Doctoral theses at NTNU, 2017:374

Stian Skjong

Modeling and Simulation of Maritime Systems and Operations for Virtual Prototyping using Co-Simulations

Doctoral Thesis

ISBN 978-82-326-2820-9 (printed version)
ISBN 978-82-326-2821-6 (electronic version)
ISSN 1503-8181

Stian Skjong

Doctoral theses at NTNU, 2017:374

NTNU
Norwegian University of
Science and Technology
Faculty of Engineering
Department of MarineTechnology

NTNU
Norwegian University of
Science and Technology

NTNU

NTNU
Norwegian University of
Science and Technology

Stian Skjong

Modeling and Simulation of Maritime Systems and Operations for Virtual Prototyping using Co-Simulations

Thesis for the degree of Philosophiae Doctor

Trondheim, December 2017

Norwegian University of Science and Technology
Faculty of Engineering
Department of MarineTechnology

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Engineering
Department of MarineTechnology

ISBN 978-82-326-2820-9 (printed version)
ISBN 978-82-326-2821-6 (electronic version)
ISSN 1503-8181

Doctoral theses at NTNU, 2017:374

MILJØMERKET 241 TRYKKSAK 731

Printed by Skipnes Kommunikasjon as

Abstract

Today, the maritime industry is facing stricter environmental requirements and a demanding market requesting greener and more advanced technology promoting lower emissions and higher fuel savings, increased operational safety and reduced operational weather dependencies. This, while the competition for new contracts forces a significant decrease in both price and "time-to-market". The Norwegian maritime industry has an edge when it comes to advanced technology and expertise in complex maritime operations. In order to keep this position a strong focus on innovation is crucial. However, the Norwegian maritime industry is also affected by high costs related to high wages and high research and development spendings. The reduced oil price, which the Norwegian maritime industry is strongly dependent on, does not help either. Hence, the Norwegian maritime industry has been forced to look for ways to reduce the high costs while at the same time maintaining a technological lead on competitors. This, through an increased focus on developing more advanced engineering technologies and methods which result in improved toolboxes that streamline work tasks and reduce both the amount of billing hours and the "time-to-market". One such technology, which has demonstrated promising properties in both the aerospace industry and the automotive industry, is virtual prototyping using *distributed co-simulations*, although bringing forth new challenges that need attention. The knowledge building project *Virtual Prototyping of maritime systems and operations* (ViProMa) was initiated in 2013 with the vision of investigating some of these challenges and to bring distributed co-simulation technology into the maritime industry. This is also the vision in this thesis.

The work presented in this thesis is carried out in the ViProMa project and is divided into three parts. The first part, consisting of Chapter 2 to Chapter 4, gives an introduction to co-simulations in general, as well as revealing some of the challenges related to using co-simulations in maritime industrial applications. The first chapter gives an introduction to the essentials in co-simulations and highlights the similarities between a general co-simulation system and a general sampled systems. Also, an introduction to both the co-simulation standard *High Level Architecture* (HLA) and the de-facto standard *Functial Mock-up Interface* (FMI) is given, where the latter is the one used in both this thesis and in the ViProMa project. An open source co-simulation software named *Coral* was developed in the ViProMa project and will be used in this thesis for conducting most of the co-simulation case studies. Also, a discussion relating system modularity, co-simulation scenario handling and configuration of co-simulations is provided.

In Chapter 3 the focus is given to the stability and accuracy of co-simulations results where the main focus is given to combining both numerical stability and dynamical stability into a joint stability analysis method for linear co-simulation systems based on subsystem discretization. This is possible since both the dynamical stability and the numerical stability in a co-simulation is highly dependent on the communication time-step size. The combined stability criterion is also applicable to non-linear systems but tend to become more conservative. Also, a general discussion of applicable dynamical stability theories are given, where the most prominent ones for linear systems are found in sampled system theory, while the most prominent ones for non-linear systems are based on system passivity considerations. The last part of the chapter presents the ECCO algorithm, a non-iterative adaptive energy-conservation-based communication time-step size controller that helps controlling the accuracy of co-simulation results. This algorithm is based on calculating the energy residuals in connections between subsystems due to the error in exchanged power between the systems introduced by sampling the connected subsystems only at given communication points.

The last chapter in Part I of the thesis gives an introduction to tightly coupled systems and possible methods of handling such systems in co-simulations. The concept of tightly coupled systems are divided into two in this thesis, namely systems that are tightly coupled through causality and systems that are tightly coupled through frequencies, where the main focus is given to the former. In general, tightly coupled systems should be implemented as one subsystem in a co-simulation since splitting them often results in problems regarding differential algebraic equations and relations. However, a method for reorganizing such algebraic relations is presented and is based on calculating the differential terms by using a low-pass filter with derivative effect. This method can also be used as a way of obtaining multiple connectivity options for subsystems in a co-simulation, which increase the possibility of connecting the subsystem to an unknown system environment. Also, the concept of *hybrid causality models* are introduced and is a special type of model switching that concerns models that have the ability to switch between inputs and outputs online during a simulation and thereby change the connectivity. Such models are useful when considering models including failure dynamics as well as when for example modeling marine power plants with weak power grid. A case study of the latter is given in the end of the chapter in order to illustrate the method, and a short discussion regarding dynamical stability

in the light of the proposed method is given.

The second part of the thesis, consisting of Chapter 5 to Chapter 7, concerns development of generic domain models which are to be used in various co-simulation case studies in Part III of the thesis. In particular, a generic marine offshore vessel model for DP-operation purposes including all relevant subsystems is derived in Chapter 5. This model also includes a crane placed on the vessel's deck and contributes also to the discussion about systems that are tightly coupled through causality. In contrast to how such systems are handled in Chapter 4 the vessel and the crane are combined into the same subsystem using the Lagrange's method, which also removes all possible differential algebraic equations. The total vessel model also includes a propulsion system, a wave-filter, a DP-control system, a crane control system and relevant environmental effects due to an irregular sea state and currents. Note that even though the total vessel model is referred to as one system in Chapter 5, it will be split into several subsystems in Part III of the thesis.

Chapter 6 presents a generic power plant model with a weak power grid, consisting of two generators powered by auxiliary diesel engines, a simple power management system and all necessary local control systems for operating the plant. Note that the generator models in this power plant are modeled as hybrid causality models, as presented in Chapter 4, in order to enable starting and stopping of arbitrarily generators online during a simulation without including a capacitive power grid effect in the total model, which also stiffens the system considerably. Even though this power plant model can be split into several subsystems in a co-simulation, as was done in Chapter 4, it has not been deemed necessary in the co-simulation case studies in Part III of the thesis since the power plant model is solved fast enough and if being split into several subsystems lower communication time-step sizes than the ones used in Chapter 8 would be required in order to provide stable co-simulation results, such as the one used in the case study in Chapter 4.

The last chapter in Part II presents a generic thrust allocation algorithm based on model predictive control (MPC) theory. This thrust algorithm is derived without including the azimuth angles in the optimization problem formulation in order to reduce the non-linearities such that the total problem formulation can be solved without implementing additional functionalities for avoiding singularities. Also, since MPC theory is used, the allocated thrusts are optimized over a time horizon instead of only in one time step. The proposed thrust allocation algorithm is benchmarked against a commonly used one-step thrust allocation algorithm and displays good characteristics. Also, the chapter has an additional focus on reducing thrust oscillations from unfiltered environmental disturbances in the measurements, which are amplified by the DP-controller, through proper tuning of the

proposed algorithm in addition to using thruster biasing. As it turns out, the power consumption from the propulsion system does not increase considerably when tuning the thrust allocation algorithm, but the oscillations in the power consumptions are significantly reduced. Also, the vessel is able to keep its position and orientation equally well in all the studied tuning cases of the thrust allocation algorithm. Note that also the proposed thrust allocation algorithm is implemented as one separate subsystem in the co-simulation case studies presented in Chapter 8.

The last part of the thesis, which consists of Chapter 8, presents four different co-simulation case studies of applications relevant in the maritime industry where the first one illustrates how co-simulations can be used for improving collaboration between researchers as a platform for connecting different work together into more complex simulators. The second case study illustrates how co-simulations can be used for optimizing system integration on a higher level, such as optimizing the interplay between the power plant in a marine vessel, the propulsion system and high level control systems such as a DP-control system. The third case study illustrates how hardware can be included into the co-simulation loop. The last case study illustrates how co-simulations can be used as an effective design tool for testing different vessel- and equipment configurations in different scenarios in high fidelity simulations in a fast and generic manner.

Chapter 9 is the last chapter in the thesis and presents some concluding remarks regarding the conducted work, as well as presenting a list of practical guidelines for performing co-simulations and a list of recommended further work.

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of Philosophiae Doctor (Ph.D.) at the Norwegian University of Science and Technology (NTNU). The research has been carried out at the department of Marine Technology (IMT), at the Marine Technology Center (MTC) at Tyholt in Trondheim from August 2014 to December 2017, with Associate Professor Eilif Pedersen (IMT) as the main supervisor and Professor Vilmar Æsøy (NTNU in Ålesund) as co-supervisor. The research is part of the project *Virtual Prototyping of Maritime Systems and Operations* (ViProMa), funded by Rolls-Royce Marine, VARD (former STX), DNV GL (former DNV) and the Norwegian Research Council, project number 225322.

Acknowledgement

My doctoral studies have for sure been an amazing, yet at times stressful and demanding journey. After finishing my MSc in Marine Technology at NTNU in 2014 I was encouraged by my supervisor Associate Professor Eilif Pedersen to apply for a PhD position at IMT, NTNU, within virtual prototyping of maritime systems and operations. I was a bit hesitant at first, but since the Norwegian maritime industry had a downturn that affected the job opportunities I applied for the position and ended up getting it. Even though I decided to never ever again take an exam at NTNU after finishing my MSc degree, I accepted the offer and had to reconcile with the fact that more exams were to be scheduled. However, I have never regretted this decision.

I would like to humbly thank my main supervisor Eilif for these three rich years as a PhD student which he has had a great influence on. He has given me valuable guidance and we have had many fruitful discussions, not only related to the topics presented in this thesis work, and his office door has always been open for me no matter how busy he has been. I would also like to thank him for always supporting me and being profoundly enthusiastic about my work. He has taught me to become a better researcher and given me many opportunities to gain more academic experience such as supervising master students and writing research project applications as well as entrusted me with the responsibility of teaching the subject Mechatronics for three consecutive years in company with PhD Candidate Børge Rokseth. I also owe my co-supervisor Professor Vilmar Æsøy from NTNU in Ålesund my gratitude. He has encouraged me and provided me with useful feedback throughout the entire process.

I would also like to thank the ViProMa project group for being attuned, focused on reaching the same goals and for supporting my research financially. In particular I would like to thank Dr. Lars T. Kyllingstad and Martin Rindarøy in SINTEF Ocean, and Dr. Severin S. Sadjina in SINTEF Ålesund for close collaborations and for all the joint work and publications produced in the project. Severin also had the postdoctoral position in the project and have together with Lars and Martin been great sparring partners. I look forward to become their colleague in SINTEF

Ocean after finishing my PhD studies.

Furthermore, I would like to thank PhD Candidate Børge Rokseth for being my sparring partner both during the master degree studies and the doctoral studies, and it has been a true privilege to share office with him all these years. In the last two years also PhD Candidate Jørgen B. Nielsen and Dr. Torstein I. Bø shared office with us and as a group we have had many interesting discussions, although often of the more philosophical kind. I would also like to thank them for various activities outside working hours, especially for all the trips to the shooting range together with Børge and Jørgen. Also, I would like to thank PhD Candidate Vladimir Krivopolianskii and PhD Candidate Endre Sandvik, who have been our office neighbors, for interesting discussions over many shared lunch meals and coffee breaks.

I can never find words to express my most profound gratitude for the love and support from my closest friends and family. I would especially like to thank my father Leif-Kristian, for teaching me the practicalities regarding engineering since before I could walk and for encouraging me to always pay attention to school, and my mother Oddhild for all the love and support and for teaching me common decency. I would also like to express my gratitude to my brother Espen and my sister Karoline and her husband for all the warmth and support. Last, but not least, I would like to humbly thank my wife Kristina who I got married to during the doctoral studies. Your love, warmth, patience and support have been crucial to reaching my goals. I am also grateful for all the warm meals and your smile I have come home to after long days at work. I would also like to apologize to you for being so caught up in work and full thinking all the time in our first year of marriage. I am deeply grateful to have you as my wife.

Stian Skjong
Trondheim, December 2017

List of Abbreviations

API	Application Programming Interface
AVR	Automatic Voltage Regulator
BSFC	Brake Specific Fuel Consumption
CSI	Common Simulation Interface
DIS	Distributed Interactive Simulation
DOF	Degree Of Freedom
DP	Dynamic Positioning
FFT	Fast Fourier Transform
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FOC	Field Oriented Controller
FOH	First Order Hold
FROH	Fractional Order Hold
HIL	Hardware In the Loop
HLA	High Level Architecture
I/O	Input-Output
IFC	Instantaneous Fuel Consumption
ISS	Input to State Stability
KKT	Karush–Kuhn–Tucker
KPI	Key Performance Indicator

LMI	Linear Matrix Inequality
MHC	Moving Horizon Control
MPC	Model Predictive Control
NLPO	Non-Linear Passive Observer
OS	One-Step
PID	Proportional-Integrate-Derivative
PMS	Power Management System
PS	Port-side
PTI	Power Take In
PTO	Power Take Off
RAO	Response Amplitude Operator
RHC	Receding Horizon Control
RK	Runge-Kutta
RTI	Run-Time Infrastructure
SB	Starboard
SISO	Single-Input-Single-Output
SoC	State of Charge
ViProMa	Virtual Prototyping of Maritime Systems and Operations
VISTA	Virtual Sea Trail by Simulation of Complex Marine Operations
VLCC	Very Large Crude oil Carrier
ZOH	Zero Order Hold

Contents

CHAPTER 1

Introduction

This chapter, which serves as a brief introduction to the topics presented in this thesis, includes a short background and a motivation for the main topics addressed in this thesis – namely the use of distributed co-simulations in the marine industry. Also, a short presentation of the ViProMa project is given before research objectives related to this thesis are defined along with a short discussion of methodology. In the end, the scope of work and the structure of the thesis are presented along with a list of relevant publications.

1.1 Background and Motivation

Today, the Norwegian maritime industrial cluster is a world leader in developing advanced customized ships and offshore vessels to the global market, particularly ships for demanding and complex operations where the price-tags often exceed 1 billion NOK. The operational systems constitute 70-80 % of the costs, and include everything from the fingertips of the operators to main propulsion power, deck machinery such as winches and cranes, cargo handling and all necessary auxiliary systems. Industrial value chains for these products are also very complex and inter-organizational, where logistics, communication and interface challenges must be handled. Project lead-times are constantly decreasing, and mistakes or system malfunctions may cause fatal incidents, project delays and costs overruns.

The ship design process has traditionally been characterized by customized solutions and strict time and resource constraints. Many of the major design variables have been decided upfront, based on experience rather than scientific principles. While scientific knowledge, methods, and tools are now highly developed and adopted within most of the involved engineering disciplines, a component and subsystem design focus is still prevalent, rather than treating ships as complete and holistic systems. It is commonly accepted that new ship designs should be optimized with respect to operational performance rather than the performance of individual components and systems, an approach which promotes the use of total vessel simulations in the design process. Simulation of total system performance will be even more important in the future where the complexity of vessel operations keep increasing. One such example is installation of heavy subsea units at several thousand meters depth and requires accuracy and control as well as a tremendous amount of power, interactions and timing. To meet performance, safety and environmental issues and cost targets, engineers must understand how the equipment will behave and if one can evaluate multiple design concepts or operational plans in an efficient manner, such as by using simulation tools, it is possible to evaluate different alternatives within short time. This leaves technology development as the predominant avenue in order to increase the operational capabilities of new designs.

Computer-based design and analysis tools for engineering purposes have developed rapidly over the last decades [1] and a wide range of specialized analysis software for e.g. structures, hydrodynamics, computational fluid dynamics, power systems and control systems are currently used in the design process to assess system performance. Multiple marine simulators do exists, such as CyberSea [2] which provides hardware-in-the-loop (HIL) testing and dynamic capability analysis, Sesam for marine systems [3] which covers risk management, encapsulating the SIMO software developed by SINTEF Ocean, the Italian Integrated Power Plant Ship Simulator [4], which analyses marine power systems, and the Marine System Simulator (MSS) [5] which is used for education and academic research purposes. A thorough review of different marine simulators is given in [6]. In addition a number of general-purpose software systems are developed, such as MATLAB/Simulink/Stateflow, Modelica/SimulationX/Dymola, 20-Sim and MapleSim. These systems combine diverse submodel objects in a single simulation. Standalone tools developed under an umbrella and aimed towards ship design, simulation, or analysis are available from ShipDesignLab@TUDelft [7] and MIT's Innovative Ship Design Lab [8]. Domain-specific analysis software for individual components and subsystems, such as power systems (PSCAD), ship motions (ShipX, VeSim [9], Shipmo, WAMIT), propulsion (AKPA, DESP), structures (RIFLEX, Flexcom3D, OrcaFlex), control systems (dSPACE) and engine systems (enDYNA, GT-Suite), are also widely available.

Both general-purpose software and domain-specific software are used during the ship design process as separate tools for design of individual systems and local system optimization. In some cases also the results from one analysis are used as initial conditions in analyses performed in other software, e.g. hydrodynamic calculations of environmental loads are used in structural analyses of the hull and for designing the propulsion system. Such analyses are often performed by spe-

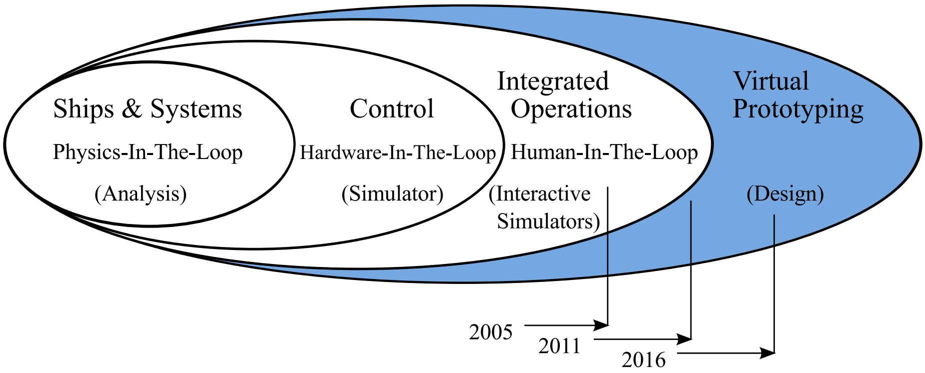


Figure 1.1: The development of system simulators for training and desing purposes in the Norwegian maritime industry

cialists in the respective fields making it more difficult to have an overall system optimization focus and the iterative process of optimizing a design becomes time consuming. One way of increasing the focus on overall system optimization as well as reducing the number of manual iterations in the design process is to integrate these software. However, this is non-trivial, due to differences in the emphasis on system modularity, model accuracy in the software and differences in software architecture, and would also required specialist training for the user.

The aviation and defence industry have developed advanced methods in both design and operation driven by customer and regulatory requirements [10, 11]. This includes the use of simulation both as a tool for operator training as well as for design purposes by implementing simulation models from a distributed collection of models and simulator sites [12]. The maritime industry has partially adopted the results of this process, most notably crew training in simulators [13] — that is, in calm water without wave effects or at a subsystem level to learn to use a special piece of equipment decoupled from the total system, where knowledge and simulation components from the design stage of a project are to a greater extent re-used [14], see Figure 1.1. Even though the use of multi-physics simulations (ship & systems), human behaviour (control) and multiple parallel marine operations (integrated operations) are demonstrated in training simulators, the use of system simulators for design is limited even though ship designers have access to advanced analysis tools. However, one notable exception is ship yards which have incorporated simulation into planning of the building process [15]. The next step in this development is the introduction of simulation based work processes (virtual prototyping), which will be a significant scientific and operational achievement for the maritime industry.

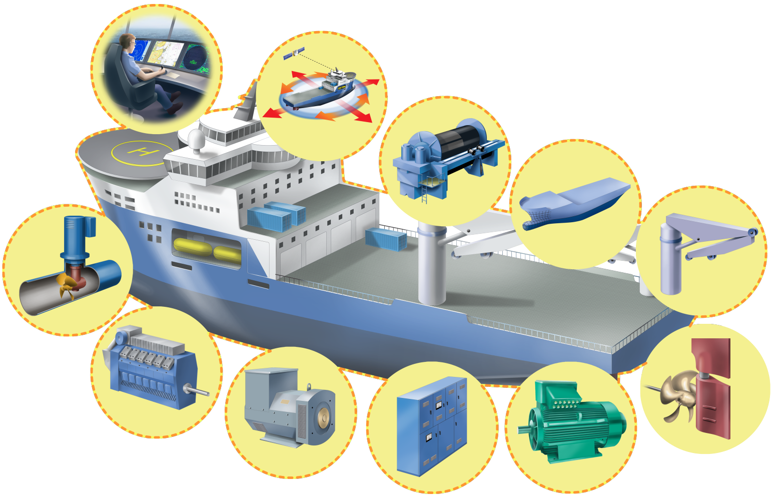


Figure 1.2: A sketch of a vessel and its subsystems as a virtual prototyping simulation model. Figure obtained from [19]

The maritime industry is not a pioneer when starting to gain interests in virtual prototyping. Both the aerospace industry and the automotive industry have been utilizing virtual prototyping methods based on simulator technology for the last decade. For example, the automotive industry has successfully incorporated simulation-based design and verification methods into their working platforms [16, 17, 18], which not only work as a local glue between different departments, but brings their third party vendors closer when researching and developing new and better products, from a component level focus to a finished product for mass production. The fact that a single product is mass produced also enables the industry to spend significant resources on optimizing both the product and the development process. This is not the case in the maritime industry where a vessel is tailored, never mass produced and where the financial surplus from a new-build is often minimal. Hence, spending resources on embedding simulation technology in the design process is a bold move and it takes the industry many years to mobilize and readjust. However, the many advantages of doing so are quite appealing in an already squeezed market and do not only facilitate shorter project lead-times but also increased possibilities, better designs and new applications for analysing performance.

Examples of such new applications include manoeuvring of ships in harsh environments, ship-to-ship operations at sea, crane operations at sea and anchor handling

[20]. These are all situations where a simulation entails a complex coupling of several disciplines as shown in Figure 1.2. By enabling such complex multi-domain simulations it is possible to analyse total vessel performance, optimize system interactions and plan for safer and more environmental friendly operations in the maritime industry.

The vision in the maritime industry is to develop a framework for overall system design, allowing configuration of ships and verification of operational performance as a part of the design process. A variety of general-purpose software and frameworks for system simulations exist, but there are no mutually adopted simulation frameworks that support total systems integration and analysis of operational performance. General software solutions for system simulations are not considered suitable for the purpose, mainly due to very time-consuming model development. Decreasing project lead-times demand rapid model development and configuration with sufficient accuracy of which general software are not suitable. The ViProMa project [21], which this thesis is a part of, aimed at investigating some of these issues and to bring the industry closer to reaching its vision by the use of *distributed co-simulations*.

1.2 The ViProMa Project

The knowledge-building project *ViProMa - Virtual Prototyping of Maritime Systems and Operations*[1] [19] was founded in 2013 by the Research Council of Norway (Grant Number 225322), NTNU, SINTEF Ocean (former MARINTEK) and the industrial partners in the project consortium consisting of VARD[2], Rolls-Royce Marine[3], and DNV GL[4]. The project aimed at integrating the current technology and know-how in the industry, and was expected to bring significant new scientific advances into the maritime industry by combining virtual prototyping and distributed co-simulation technology with the de-facto standard *Functional Mock-up Interface* (FMI) in its core, see Chapter 2. This, in order to enable new, safer and greener solutions, design concepts and equipment combinations to be simulated and tested in a laboratory environment before being built, and to encourage rapid innovation and help bring design, training and operations closer together in the industry.

The composition of disciplines has been crucial for the success of the project, ranging from hydrodynamics, mathematical modeling, control and simulation expertise, experience with maritime systems and operations, computer science and

[1] www.viproma.no

[2] www.vard.com

[3] www.rolls-royce.com

[4] www.dnvgl.com

software architecture, and numerics, just to mention a few. This has forced the project group to keep the research and development in the project as generic as possible, resulting in the fact that most of the research results also apply directly to other disciplines, not only being limited to the maritime industry.

The project was finished by the end of 2016, but the results from the project have benefited other project, such as *SFI Smart Maritime* [22] and *VISTA - Virtual Sea Trail by Simulation of Complex Marine Operations* [23], hence, the project vision and further software developments are continued in other existing project and possibly through future spin-off projects, some already being under development at this stage.

The work presented in this thesis is financially supported by the ViProMa project, where the use of distributed co-simulations in the maritime industry is central, and is concentrated around the research objectives presented in the following.

1.3 Research Objectives and Methods

The main objective for the work presented in this thesis is to investigate how different domain models from different software can be integrated in a co-simulation environment, the gained opportunities and technological advantages this may bring forth, and to identify possible related challenges and potential solutions. This, in order to facilitate the development of safer, greener and more complex offshore vessels more efficiently in the future. Hence, applied research will be emphasized where possible. This main objective is divided into smaller research objectives given in the following:

RO1 Investigate possible challenges related to simulating typical maritime systems and operations as distributed co-simulations. In particular, investigate how the communication frequency between subsystems in co-simulations affects the overall numerical stability and the accuracy.

RO2 Investigate different challenges related to splitting systems that are strongly dependent on each other into subsystems for co-simulation purposes. In particular, investigate methods for removing differential algebraic dependencies between subsystems in co-simulations.

RO3 Develop essential generic mathematical domain models as a base for assembling total offshore vessel simulators suited for analysing different vessel operations. In particular, develop a generic vessel model and models for the most relevant subsystems, e.g. relevant vessel control systems, propulsion system and energy systems.

RO4 Evaluate the use of co-simulations in typical maritime applications by applying co-simulations in relevant case studies, which also demonstrate how to solve possible co-simulation related challenges. Also, develop an open-source vessel simulator that is solved by the ViProMa project's open source co-simulation master algorithm *Coral*, for both demonstrator purposes and for providing a generic vessel simulator framework as a foundation for further research and development.

Some of these research objectives are more closely linked than others, such as **RO1** and **R02** where splitting systems that are strongly dependent on each other also affects the stability and the accuracy of the simulation results. In general, **RO1** and **RO2** treat more fundamental topics related to distributed co-simulations and might give a foundation for **R03** and **RO4**, which on the other hand are more application oriented objectives.

In the following, a short discussion of research methods are given as well as a short introduction to *bond graphs*, a graphical modeling methodology that is essential for some of the work presented in this thesis.

1.3.1 Methodology

In this work, modeling, control and stability theory are central topics and, thus, devoted much attention. Modeling theory, the theory of mathematically representing realistic behaviours or effects from physical systems, economics or social situations and scenarios through equations, is presented in many academic disciplines and fields of engineering. Hence, the presentation of modeling theory may vary, mostly because of model fidelities, the intentions of the model and which mathematical tools and analyses being in focus. Here, the modeling theory is limited to include only mathematically representations of physical systems and dynamical effects, normally represented by differential equations, differential algebraic equations and/or empirical algebraic relations. When it comes to rigid body systems the use of Lagrange's method [24] is a prevalent energy-based method for combining kinetic and potential energy in a system, and results often in differential equation representations. The mathematical system representation can either be implemented directly on the equation level through a suited programming language, or by the use of suited modeling software utilizing predefined iconic models, block diagrams or bond graphs [25, 26, 27].

One of the many intentions for making a mathematical model representation of a physical system is to be able to analyse the system behaviour and to manipulate the characteristics through proper control of the system. This can be achieved through the use of *off-the-shelf* controllers such as PID-control laws, or through model-based control law designs [28] based on the most significant system dynamics and on suited dynamic stability theory. Dynamical system stability can be evaluated by the use of many different methods, for example by using methods based on the eigenvalues in linear systems [29], which are measures of energy dissipation, or through methods using functions that represent the energy in the system, such as *Lyapunov functions* [30], which is usually the way of assuring stability when handling non-linear dynamics. When it comes to stability in simulations, numerical stability is as important as dynamical stability and has a significant impact on the accuracy in the simulation results. Fortunately, the numerical stability in a simulation is as for the dynamical stability dependent on the system dynamics, the eigenvalues in the system that is to be solved [31, 32].

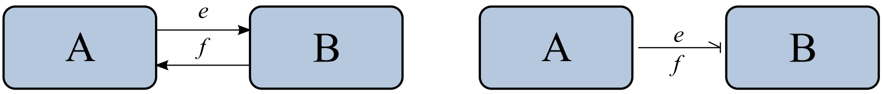
Since modeling, control and stability theory have strong relations to energy methods it is a good approach to base the co-simulation theory on energy methods as well. When co-simulations consist of connected subsystems representing physical systems, the connections between the subsystems should be properly defined based on aspects of energy, which also relates to how the different subsystems are being modeled. The idea of connecting different subsystems through energy is one of the core ideas in bond graph modeling theory, where subsystems are connected through the exchange of power. The ViProMa project goes as far as recommending the use of *power bonds* as a high level interface between subsystems in a co-simulation whenever possible. Even though bond graphs are not directly in the scope in this work, a short introduction to bond graph theory is given in the following.

1.3.2 Bond Graph Modeling Theory

In bond graph theory each dynamical effect in a model is coupled through the exchange of energy. This, because all physical systems, independent on energy domains, have the same definition of energy. The exchange of power is represented through two variables in bond graph theory denoted *effort*, e, and *flow*, f. These two variables are referred to as *power variables* since the product of them is power, $P\ [W]$ and the total energy transfer between two subsystems, $E\ [J]$, as the integral of power,

$$\begin{aligned} P &= e \cdot f \\ E &= \int_0^t P dt = \int_0^t e \cdot f dt \end{aligned} \tag{1.1}$$

Figure 1.3(a) illustrates the power connection between two subsystems. In bond graph theory the two arrows connecting the two systems in Figure 1.3(a) are replaced with one *power bond*, as shown in Figure 1.3(b), and gives an equivalent



(a) Block diagram connection

(b) Equivalent power bond connection

Figure 1.3: Subsystem A and B connected by the power variables e and f

Table 1.1: Power variables in bond graph theory

Energy domain	**Effort**	**Effort units**	**Flow**	**Flow units**
Mechanical (trans.)	Force	N	Linear velocity	m/s
Mechanical (rot.)	Torque	Nm	Angular velocity	rad/s
Electrical	Electromotive force	V	Current	A
Hydraulic	Pressure	Pa	Volumetric flow rate	m^3/s
Thermal	Temperature	K	Entropy flow rate	W/K

representation of the coupling between the systems. Note that the power bond has an half arrow and a orthogonal line in the end of the connection. The half arrow illustrates the positive power direction while the orthogonal line, the *causality stroke*, illustrates that the effort is set by subsystem A.

Even though the definition of power is independent of the energy domain in a system, the power variables are different. In a mechanical system with only linear translations, the effort has the quantity of force [N] and the flow the quantity of linear velocity [m/s]. Table 1.1 gives an overview of the different power variables in the most common energy domains.

In bond graph theory each dynamical effect in a system is represented by one of the nine basic bond graph elements given in Table 1.2. Note that only two of these elements, the mnemonic elements $\boldsymbol{C}$ and $\boldsymbol{I}$, are potential candidates for producing differential equations and states. The $\boldsymbol{C}$-element is associated with compliance, storing of potential energy, and can for example represent the dynamics of a spring in mechanical systems, a capacitor in electrical systems or an accumulator in hydraulic systems. The state is often denoted q and is defined as the integral of the flow. The $\boldsymbol{I}$-element is associated with inertia, storing of kinetic energy, and can for example represent the dynamics of a mass in mechanical systems, an inductance in electrical systems or fluid inertia in a hydraulic system. The state is often denoted p and is defined as the integral of effort. Note that p is often referred to as the momentum in mechanical systems.

The seven other basic bond graph elements contribute with algebraic equations. Two of the elements are characterized as source elements, namely $\boldsymbol{S_e}$ and $\boldsymbol{S_f}$,

Table 1.2: Basic bond graph elements.

Graph	Relation	Graph	Relation	Description
$S_e \longrightarrow\!\mid$	$e = e(t)$, given	$S_f \longmapsto$	$f = f(t)$, given	Sources
$\longmapsto \text{R}$	$e = \Phi_R(f)$	$\longrightarrow\!\mid \text{R}$	$f = \Phi_R^{-1}(e)$	Dissipator
$\longmapsto \text{C}$	$e = \Phi_C^{-1}(\int_0^t f dt)$	$\longrightarrow\!\mid \text{C}$	$f = \frac{d}{dt}[\Phi_C(e)]$	Compliance
$\longrightarrow\!\mid \text{I}$	$f = \Phi_I^{-1}(\int_0^t e dt)$	$\longmapsto \text{I}$	$e = \frac{d}{dt}[\Phi_I(f)]$	Inertia
$\overset{e_1}{\underset{f_1}{\longmapsto}} \text{TF} \overset{e_2}{\underset{f_2}{\longmapsto}}$	$e_1 = me_2$ $f_2 = mf_1$	$\overset{e_1}{\underset{f_1}{\longrightarrow\!\mid}} \text{TF} \overset{e_2}{\underset{f_2}{\longrightarrow\!\mid}}$	$e_2 = \frac{1}{m}e_1$ $f_1 = \frac{1}{m}f_2$	Transformator
$\overset{e_1}{\underset{f_1}{\longmapsto}} \text{GY} \overset{e_2}{\underset{f_2}{\longrightarrow\!\mid}}$	$e_1 = mf_2$ $e_2 = mf_1$	$\overset{e_1}{\underset{f_1}{\longrightarrow\!\mid}} \text{GY} \overset{e_2}{\underset{f_2}{\longmapsto}}$	$f_2 = \frac{1}{m}e_1$ $f_1 = \frac{1}{m}e_2$	Gyrator
$\begin{tikzcd} & {} & \\ {} \arrow[r, "e_1", "f_1"'] & \mathbf{1} \arrow[u, "e_2", "f_2"'] \arrow[r, "e_3", "f_3"'] & {} \end{tikzcd}$	$e_1 - e_2 - e_3 = 0$ $f_1 = f_2 = f_3$	$\begin{tikzcd} & {} & \\ {} \arrow[r, "e_1", "f_1"'] & \mathbf{0} \arrow[u, "e_2", "f_2"'] \arrow[r, "e_3", "f_3"'] & {} \end{tikzcd}$	$e_1 = e_2 = e_3$ $f_1 - f_2 - f_3 = 0$	Nodes

providing an effort and a flow as a source, respectively, and the $\boldsymbol{R}$-element handles the dissipation of energy in a system. When connecting energy domains in bond graph theory the elements $\boldsymbol{TF}$ and $\boldsymbol{GY}$ are quite useful. The $\boldsymbol{TF}$-element is a transformer element that multiplies both the effort and the flow with a transformer modulus. The $\boldsymbol{GY}$ element is slightly different than the transformer element, it multiplies a gyrator modulus by a flow to produce an effort, and the other way around as shown in Table 1.2. The last two basic elements are two junctions, namely $\mathbf{0}$ and $\mathbf{1}$, where $\mathbf{0}$ sums flows around its node and $\mathbf{1}$ sums efforts around its node. These two junction elements connect all the other elements such that a total bond graph can be established for a system. Various introductory examples of bond graph models are given in the literature and the reader is referred to [25] for a thorough review of the modeling theory beyond this short presentation.

1.4 Scope of the Work

The work presented in this thesis focus on the use of co-simulations in the maritime industry with emphasis on applicable results and practical usage where possible,

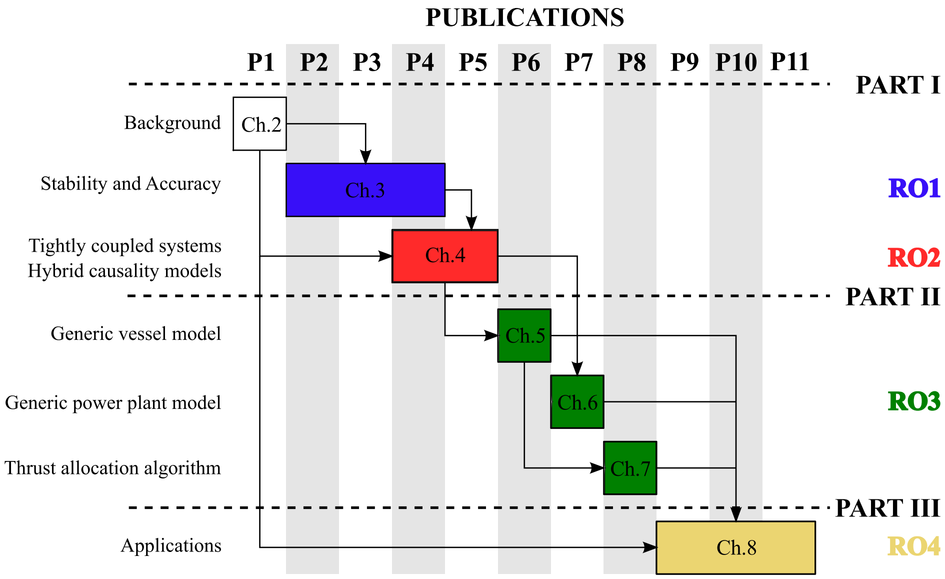


Figure 1.4: Structure of the thesis linking publications, chapters and research objectives together

although the research regarding stability and accuracy in co-simulations, as well as tightly coupled systems, will require a more theoretical approach.

The thesis is divided into three different parts. The first part, *Co-Simulation Theory*, gives an introduction to co-simulations as well as studying numerical stability, accuracy and tightly coupled systems regarding co-simulations. The latter also result in a framework for one special type of switched models, namely *hybrid causality models*. The second part, *Modeling and Control*, presents different generic domain models and control systems such as a generic vessel model, including e.g. a deck crane, propulsion system and DP-control system, a generic power plant model and a thrust allocation algorithm. The last part, *Applications of Co-Simulations*, presents different case studies that show different applications of co-simulations relevant in the maritime industry.

In short, Part I gives an introduction and a background in co-simulations and related challenges while Part II presents generic mathematical models needed for constructing suited co-simulation case studies, that are designed to highlighting the possible challenges and opportunities related to co-simulations, and for evaluating the use of co-simulations in the maritime industry. These case-studies are presented and analysed in Part III. A schematic overview of the parts, chapters, publications and research objectives are given in Figure 1.4. Note that **P1-P11** denote publications included in this thesis that are listed in section 1.6. Also, the research objectives are as presented in section 1.3 and are given in different color codes in order to illustrate in which part- and in which chapter(s) of the thesis the research objectives are treated.

In the following, a more detailed outline of the thesis, including a short summary of each chapter, is presented.

1.5 Thesis Outline

The chapters are structured in a chronological sequence where the theory presented in previous chapters may provide a basis for the topics discussed in the following chapters, as shown in Figure 1.4. A brief description of each chapter is provided as follows:

Part I - Co-Simulation Theory

Chapter 2: This chapter includes a short presentation and comparison of distributed systems, distributed simulations and distributed co-simulations, as well as an introduction to two co-simulation standards; the HLA standard and the de-facto standard FMI, where the latter will be used in this thesis. Also, a short discussion of the similarities between a general sampled system and a co-simulation system is given, a presentation of the open source co-simulation software *Coral*, developed in the ViProMa project, and the open source vessel simulator developed in this thesis will be given. In the end of the chapter the concept of system modularity, configuration of co-simulations and co-simulation scenario control will be discussed and related to each other.

Chapter 3: This chapter includes a presentation of stability in co-simulations, with main focus on numerical stability, as well as a short discussion about co-simulation accuracy. A literature review of relevant dynamical stability theories is conducted, ending in a more thorough study of linear co-simulation systems. A combined stability analysis method for assuring both numerical and dynamical stability in co-simulations is also presented. In the end of the chapter, an adaptive communication time-step size controller is presented, which improves co-simulation accuracy while maximizing the communication time-step size according to pre-set accuracy tolerances.

Chapter 4: This chapter treats the problem of tightly coupled systems in a co-simulation, with main focus on tightly coupled causality systems. A method for reformulating differential algebraic equations is presented and analysed. This method leads to the option of having hybrid causality models, which extend the connectivity range as well as enabling modeling of failure dynamics in a system.

The method is illustrated with an example of a marine power plant having a weak power grid. The proposed method also has nice properties when it comes to analysis of dynamical stability which is also illustrated through the marine power plant example.

Part II - Modeling and Control

Chapter 5: This chapter presents a vessel model including a deck crane, which together is considered tightly coupled. The vessel and the crane model are combined by using Lagrange's method, including quasi-coordinates, and power variables. A deep-going case study of a marine vessel with a deck crane, including models for all relevant subsystems is considered and two different crane loading cases are compared – one including payload attached to the crane through a wire, and the other without any payload. This vessel model is used in later chapters in various co-simulation case studies, but where some of the vessel systems are considered as separate subsystems in the co-simulation.

Chapter 6: This chapter presents a complete marine power plant model including two generators powered by auxiliary diesel engines, a simple power management system and all relevant control systems needed for operating the plant. This model is to be used in co-simulations in Chapter 8. The generator models are hybrid causality models, meaning that it is possible to start and stop arbitrary generators in the power plant. Much focus is given to marine power plant control, such as voltage control, frequency control, generator synchronization and active- and reactive power sharing. Also, numerical stability of the hybrid generator models are discussed based on the causality orientation of the models. A case-study is presented and illustrate the stability of the generator models as well as the properties of the presented control systems.

Chapter 7: This chapter presents an optimization based thrust allocation algorithm based on Model Predictive Control (MPC) theory. The proposed algorithm does not contain any thruster azimuth angles since all thrust forces are represented as vectors. Hence, the algorithm is simpler to solve than when including the azimuth angles in the optimization problem formulation. The algorithm is benchmarked against a commonly used thrust allocation algorithm and a study of optimal cost function weights as well as the length of the optimization horizon is performed. The main case study is presented in the end in the chapter where the focus is given to how to tune the algorithm in order to reduce thrust force oscillations and thereby reduce oscillations in propulsion system's power consumption.

Part III - Applications of Co-Simulations

Chapter 8: This chapter addresses applications of co-simulations in the maritime industry. Four different case studies are presented to illustrate possible applications of co-simulation technology in the maritime industry. These case studies include collaboration between researchers using co-simulations, Hardware-In-the-Loop (HIL) in co-simulations, optimizing system integration using co-simulation and testing different vessel configurations using co-simulation. Much focus is devoted to the case study presented in 8.3 since it contains an open source vessel simulator derived in this thesis.

Chapter 9: This chapter summarizes the final conclusions for the work presented in this thesis, as well as giving some practical guidelines for conducting distributed co-simulations. The chapter ends with presenting recommendations for future work.

1.6 Publications

This thesis is based on results that are either published or submitted for publication, some of them in collaboration with colleagues. The main publications that are included in this thesis, as well as other publications published during this thesis work, are listed in the following.

Publications included in the Thesis

[33] (**P1**) Severin S. Sadjina, Lars T. Kyllingstad, Martin Rindarøy, Stian Skjong, Vilmar Æsøy, Dariusz Fathi, Vahid Hassani, Trond Johnsen, Jørgen B. Nielsen, Eilif Pedersen. *Distributed Co-Simulation of Maritime Systems and Operations*. Submitted for publication, 2017.

[34] (**P2**) Stian Skjong, Eilif Pedersen. *On the Numerical Stability in Dynamical Distributed Simulations*. Submitted for publication, 2017.

[35] (**P3**) Severin S. Sadjina, Lars L. Kyllingstad, Stian Skjong, Eilif Pedersen. *Energy conservation and power bonds in co-simulations: non-iterative adaptive step size control and error estimation*. Published in Springer - Engineering with Computers, Vol. 33 (2017), Issue 3, pp. 607–630.

[36] (**P4**) Stian Skjong, Eilif Pedersen. *The Theory of Bond Graphs in Distributed Systems and Simulations*. Published in ICBGM'16 Proceedings of the International Conference on Bond Graph Modeling and Simulation. Society for Computer Simulation International (SCS). pp. 147-156.

[37] (**P5**) Stian Skjong, Eilif Pedersen. *Hybrid Causality Model Framework in-*

cluding Explicit Reformulation of Differential Algebraic Equations using Perturbations corresponding to added System Dynamics. Submitted for publication, 2017.

[38] (**P6**) Børge Rokseth, Stian Skjong, Eilif Pedersen. *Modeling of Generic Offshore Vessel in Crane Operations With Focus on Strong Rigid Body Connections*. Published in IEEE Journal of Oceanic Engineering, Vol. 42 (2017), Issue 4, pp. 846–868.

[39] (**P7**) Stian Skjong, Eilif Pedersen. *A Real-Time Simulator Framework for Marine Power Plants with Weak Power Grids*. Elsevier Journal of Mechatronics, Vol. 47 (2017) pp. 24–36.

[40] (**P8**) Stian Skjong, Eilif Pedersen. *Non-angular MPC-based Thrust Allocation Algorithm for Marine Vessels - A Study of Optimal Thruster Commands*. Published in IEEE Transactions on Transportation Electrification (2016). Vol. 3, Issue 3, pp. 792–807.

[41] (**P9**) Stian Skjong, Eilif Pedersen. *Co-Simulation of a Marine Vessel in DP-Operations including Hardware-In-the-Loop (HIL)*. Proceedings of the ASME 2017 36th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2017, June 25-30, Trondheim, Norway.

[42] (**P10**) Stian Skjong, Martin Rindarøy, Lars T. Kyllingstad, Vilmar Æsøy, Eilif Pedersen. *Virtual Prototyping of Maritime Systems and Operations - Applications of Distributed Co-Simulations*. Accepted for publication in Journal of Marine Science and Technology, 2017.

[43] (**P11**) Kevin K. Yum, Stian Skjong, Bhushan Tasker, Eilif Pedersen, Sverre Steen. *Simulation of a Hybrid Marine Propulsion System in Waves*. Proceedings of 28th CIMAC World Congress (2016), Helsinki, Vol. 202.

Additional Publications not included in the Thesis

[28] Stian Skjong and Eilif Pedersen. *Model-based control designs for offshore hydraulic winch systems*. Elsevier Journal of Ocean Engineering, Vol. 121, pp. 224–238, July 2016.

[21] Vahid Hassani, Martin Rindarøy, Lars T. Kyllingstad, Jørgen B. Nielsen, Severin S. Sadjina, Stian Skjong, Dariusz E. Fathi, Trond Johnsen, Vilmar Æsøy, Eilif Pedersen. *Virtual Prototyping of Maritime Systems and Operations*. Proceedings of the ASME 2016 35th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2016, June 19-24, Busan, South Korea.

[44] Anna Swider, Stian Skjong and Eilif Pedersen. *Complementarity of Data-Driven and Simulation Modeling Based on the Power Plant Model of the Offshore Vessel*. Proceedings of the ASME 2017 36th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2017, June 25-30, Trondheim, Norway.

[45] Thomas Evang, Stian Skjong and Eilif Pedersen. *Crane Rig: An Experimental Setup for Developing and Verifying New Control Methods for Marine Crane Operations*. Proceedings of the ASME 2017 36th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2017, June 25-30, Trondheim, Norway.

Other Material

[19] ViProMa project group. *The web-page for the ViProMa project*, *https://viproma.no*, 2017.

Part I

Co-Simulation Theory

The scientist discovers a new type of material or energy and the engineer discovers a new use for it.
- Gordon Lindsay Glegg, *The Development of Design* (1981)

CHAPTER 2

Background in Distributed Co-Simulation

This chapter is based on some of the topics presented in [33, see **P1** in section 1.6], together with a presentation of co-simulations in general, standards and the co-simulation software *Coral*. Co-simulation systems are similar to sampled systems where both the sampling frequency and the sampling method play significant roles when it comes to both system stability and quality of simulation results. Hence, an introduction to sampled systems are given in order to both highlight the similarities between a general sampled system and a co-simulation system as well as to provide a baseline for the topics presented in Chapter 3. In the end of this chapter a short discussion relating system modularity, simulation scenario control and configuration of co-simulations will be presented. Also, the discussion will focus on highlighting possible challenges related to subsystem modularity and scenario handling when using co-simulations as a design tool including multiple subsystem developers and different subsystem fidelities.

2.1 Introduction and Motivation

The concept of distributed systems maybe familiar to most people, although the associations and perhaps the definitions may vary slightly across disciplines. However, the consequences of distributing systems are coinciding in most disciplines and involve separating coupled systems while still maintaining some type of connections, even though these normally are restricted. This is also why the concept of distributed systems is not defined when considering physical systems, which always are either continuously connected or not connected at all. Hence, distributed systems can not be defined solely based on continuous time considerations. In the field of mathematics and computer science these distributed system connections are normally defined as discrete events where the subsystems can exchange data in a structured manner and where both the frequency of data exchange and the method for handling the exchange data can have significant impact on the performance of each connected subsystem, as will be discussed in more detail in section 2.2.1.

In this chapter the main focus will be given to distributed systems in the light of modeling and control theory. For the readers familiar with general modeling theory the concept of distributed systems may invoke different associations, for example related to discretization of physical systems such as lumped system approximations, finite element methods and control volume modeling approaches, and perhaps even numerical solver theory for solving continuous differential equations and partial differential equations, and problems related to systems that are strongly dependent on each other through differential algebraic relations, mathematically speaking. Even though theory of discretizing physical systems will not be discussed in any detail here it provides a good background for the readers familiar with it. On the other hand, numerical solver theory and differential equations will be given much focus in Chapter 3 and distributed systems that are dependent on each other through differential algebraic relations will be discussed in Chapter 4. For readers that are versed in general control theory distributed systems might trigger associations to general sampling theory, discrete control systems and algorithms. As a matter of fact, distributed systems are closely related to sampled systems and this link will be discussed in more detail in section 2.2.1.

To avoid confusions and to provide a common baseline for the presentation of distributed co-simulations, a distributed system is defined in the field of computer science in the following.

Definition 2.1 (Distributed System [46]). *A distributed system is a collection of independent computers that appear to the user of the system as a single computer. The computers do not have any form of shared memory, and communication happens on a message-only basis.*

Note that the definition does not explicitly relate distributed systems to discrete time events, but is implied through the use of multiple computers which only do calculations in a discrete manner and hence, are also restricted to exchange information on specific discrete time events. Based on Definition 2.1, one can understand that people from different disciplines put different meaning in it, especially when mixing it with *distributed simulations,* as pointed out in Remark 2.1.

Remark 2.1 (Distributed Simulation [47]). A distributed simulation is often understood as a sub-domain of distributed systems and deals with software that is executed in parallel on multiple computers.

Since a computer is such a wide term, ranging from micro-controllers and chips to personal laptops and stationary computers, and since software is any form of executable machine code, distributed systems and simulations can be found all around us in everyday systems, e.g. modern transportation systems such as cars, aeroplanes and ships, where many different computers are connected and execute tailored software, often dependent on system measurements, in such a way that the combined distributed system enhances the performance and the safety of operating the system. On the other hand, distributed systems can also be found in more mathematical disciplines where distributing a system is done in order to enhance the available computational power, to *share* the computational loads. One example of such is weather calculations which is a complex and a computational expensive procedure [48]. As a matter of fact, a modern computer containing multiple processors is closely related to a distributed system itself. Although distributed systems include many different types of systems and system configurations, they all have one thing in common, namely the discrete communication points between the computing members. Even though these discrete communication points are the enablers of distributed systems and simulations, they are also a curse when it comes to simulation accuracy, which is discussed in more detail in Chapter 3.

Distributed system technology is a huge topic within computer science and, hence, a thorough review of distributed systems containing in-depth details will not be given any particular attention here, and the reader is referred to [33] where some aspects are discussed. Also, a good introduction to distributed systems and related topics from a theoretical perspective can be found in [49] while a more practical application oriented presentation can be found in [50].

While distributed simulation technology maybe associated with the intention of distributing computational loads, the possibility to link different software and hardware, that may run with different operative systems, in a single simulation environment is at least equally important. These two features of distributed systems and simulations can be seen as the main reason for distributing a system or for performing a distributed simulation, and combined they enable favourable advantages that exceed by far the possibilities associated with running software on a single computer. This is especially the case in the industry where tailored software are used for different engineering disciplines, often related to complex and computational demanding calculations, e.g. calculating hydrodynamic loads on a maritime vessel. By taking advantage of distributed simulations one can connect these specialized software in a generic way such that the scope of the simulation gets expanded to include the entire system, not only individual subsystems, which enables analysis of the total system behaviour. Here, such a total system could for example be an entire ship with all its subsystems, ranging from power plant, propulsion system and deck machinery to global control and power management systems. Even though many of these subsystems are complex and computational demanding it is still possible to solve the entire connected system because of the computational load sharing possibilities the distributed simulation technology brings forth.

In some simulations the need for additional computational power is not the main issue, but connecting different software and hardware into a combined simulation. Such simulations are referred to as *co-simulations* and is the main simulation method used in this thesis.

2.2 Co-Simulation Essentials

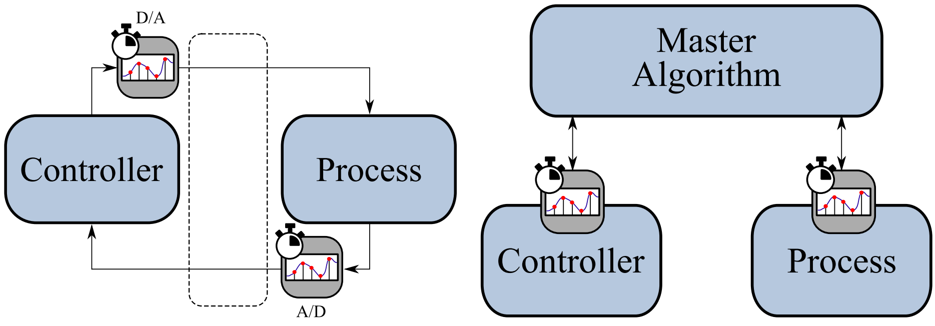
A thorough review of co-simulations and closely related topics are given in [51, 52] and, thus, only a short presentation grasping the essence of co-simulations is given here. Even though the terms co-simulations and distributed simulations are often mixed in the literature, there is a small, but distinct difference. In comparison to Definition 2.1 and Remark 2.1, a co-simulation is defined as follows.

Definition 2.2 (Co-Simulations [33]). *Co-simulation is a simulation technique in which the computations associated with different subsystems are performed independently from each other, and the exchange of data between subsystems is restricted to discrete* communication points *(sometimes called synchronization points). Each subsystem is then free to use the solver strategy and internal local time step size (also referred to as micro time step size) which is deemed most suited. The time between communication points, the global communication time steps (also referred to as macro time steps), will generally be significantly longer than the local time steps (micro time steps) of most subsystems.*

This means that the definition of co-simulations does not state anything about the use of multiple connected computers, which is the core idea in the definition of distributed simulations. However, a co-simulation can also be a distributed simulation as specified in Remark 2.2.

Remark 2.2. A co-simulation is said to be a distributed simulation if different parts of the simulation are running on different processors or on different computers in a network in order to distribute the workload between them.

In this work the co-simulations are performed on one single computer but where the subsystems in the co-simulations are treated as separate processes solved either on one processor or multiple processors, depending on the computational loads. Also, most of the theory presented in this thesis applies both to co-simulations and distributed simulations. Hence, the two terms will be used interchangeably in a consistent manner in this work.



(a) Controlled process **(b)** Co-simulation system

Figure 2.1: Comparison of a sampled process controlled by a digital controller (**a**), and a co-simulation of the same system (**b**). Note that D/A in (**a**) denotes digital to analogue signal conversion and A/D denotes analogue to digital signal conversion

As mentioned earlier, co-simulations are similar to sampled systems since the subsystems in a co-simulation exchange information on given discrete communication points as illustrated in Figure 2.1. Note that the dashed rectangle in Figure 2.1(a) is comparable to the master algorithm in Figure 2.1(b) except that it does not control the local propagating times in the controller and the process. Also, all signal conversion, both D/A and A/D happens locally in each subsystem in a co-simulation and is determined by the co-simulation standard, see section 2.3, and the communication frequency. In other words, the subsystem outputs in a co-simulation get sampled on demand by the co-simulation master algorithm, which not only controls the global simulation time but also the exchange of information between the subsystems, as will be elaborated in section 2.2.2. The sampling theory itself brings useful tools that can be applied to a co-simulation system which not only help to analyse the system but also to understand the special characteristics of a co-simulation system and provides an important foundation for the work presented in the following chapters. Hence, a discussion relating co-simulations to sampling theory is given in the following.

2.2.1 Co-Simulations and Sampling

At each communication point in a co-simulation, which is by definition a discrete event, the subsystems in the co-simulation send their respective outputs and receive new inputs from the master algorithm. This data exchange is similar to making measurements on physical systems where physical sensors, e.g electrical and/or mechanical sensors, are used to measure the system characteristics. These measurements are normally digital samplings of the system.

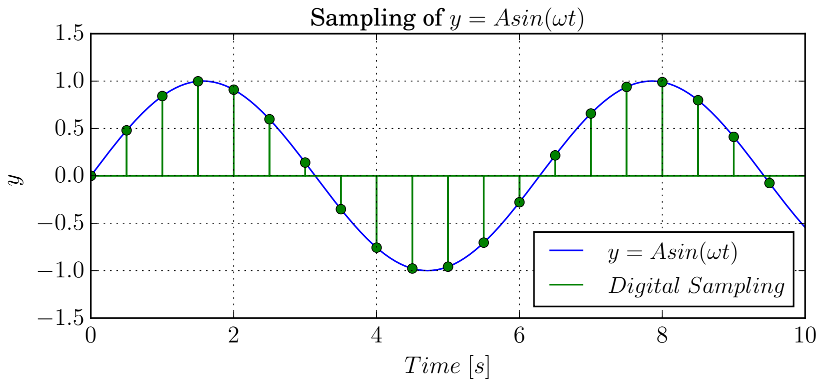


Figure 2.2: Digital sampling of $y = Asin(\omega t)$, T_d =0.5 s, A = 1.0, ω =1.0 $^{rad}/_{s}$.

Digital Sampling

In general, a digital sampling y_s of a subsystem output y can be expressed as an impulse sampling [53] and given as

$$y_s(t) = y(t) \sum_{n=-\infty}^{n=\infty} \delta(t - nT_d) \tag{2.1}$$

where T_d is the sampling time-step size, n is the sampling number and $\delta(t - nT_d)$ is the Dirac-Delta pulse function,

$$\delta(t - nT_d) = \begin{cases} \infty & \text{for } t = nT_d \\ 0 & \text{else} \end{cases} \tag{2.2}$$

having the property

$$\int_{-\infty}^{\infty} \delta(x)dx = 1 \tag{2.3}$$

A typically digital sampling of a signal is shown in Figure 2.2. As the figure shows, the sampled signal only consists of sampling impulses and is zero between each communication point in a co-simulation. Hence, the sampled digital signal must be converted to an analogue signal such that it becomes at least piecewise continuous for the subsystems receiving the signal. Two common methods are Zero-Order-Hold (ZOH) and First-Order-Hold (FOH).

Digital to Analogue

The ZOH method keeps the input signal to a slave constant between the communication points and the sampled signal looks like a stair-case when compared to

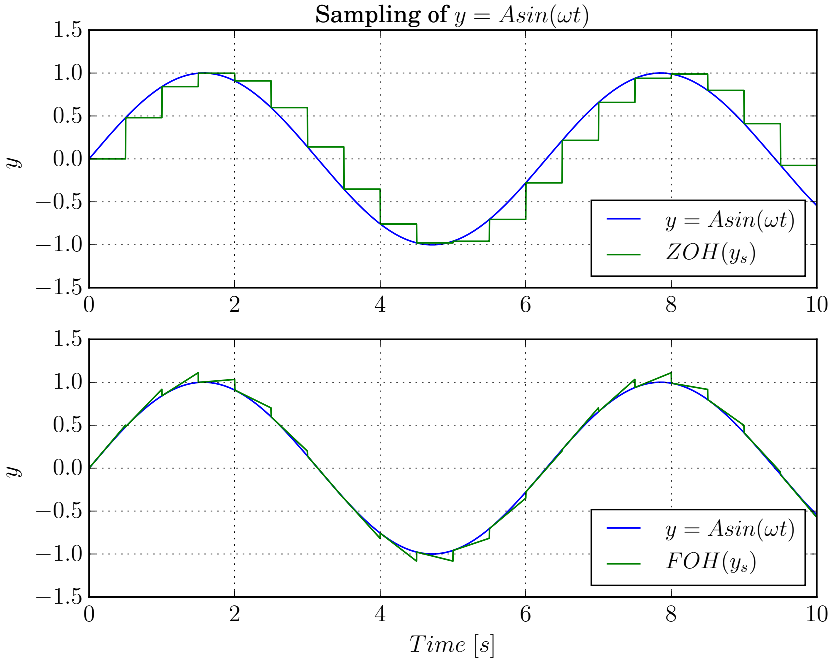


Figure 2.3: Comparison of Zero Order Hold (ZOH) and First Order Hold (FOH) when sampling $y = Asin(\omega t)$, T_d =0.5 s, A = 1.0, ω =1.0 $^{rad}/_{s}$.

the original signal. Mathematically, the ZOH can be expressed as

$$y_{ZOH}(t) = y_s(nT_d) \tag{2.4}$$

for $nT_d \leqslant t < (n+1)T_d$. The corresponding transfer function for the ZOH [54] is given as

$$H_{ZOH}(s) = \frac{1 - e^{-sT_d}}{s} \tag{2.5}$$

The FOH method, in comparison to the ZOH method, use the rate of the sampled signal as well [55]. Figure 2.3 shows a simple comparison between the two methods where a $\sin(\cdot)$ function is sampled with the two methods having the same sample frequency. From the figure it seems like the FOH method outperforms the ZOH method when it comes to replicating the sampled signal. This is mostly true in all cases according to [56] which compares the two methods, considering non-linear systems which also are affected by signal delays. The figure also shows that the FOH method displays sawtooth characteristics of the outputted signal. This can be improved by combining the ZOH and FOH into another method called *Fractional Order Hold* (FROH) [57]. Nevertheless, the co-simulation master algorithm

Coral [33] (see section 2.4.1) does only support ZOH at this point. Higher order hold functionality has of yet not been deemed necessary in the ViProMa project because many of the model libraries used in the project do not support this. Hence, the focus here is given to ZOH.

When subsystems in a co-simulation are sampled, they exchange data according to the mapping of the subsystems in the co-simulation and is controlled by the co-simulation master algorithm. A general introduction to such an algorithm is given in the following.

2.2.2 General Co-Simulation Master Algorithm

To illustrate the use of co-simulations in practise, consider two subsystems each expressed as

$$\begin{aligned} \dot{\boldsymbol{x}}_i &= \boldsymbol{f}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{\tau}_{c,i}) \\ \boldsymbol{y}_i &= \boldsymbol{h}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{\tau}_{c,i}) \end{aligned} \tag{2.6}$$

where $\boldsymbol{x}_i \in \mathcal{R}^n$ is the state vector for subsystem i, $\boldsymbol{u}_i \in \mathcal{R}^m$ is the input vector from the connected subsystems for subsystem i, $\boldsymbol{\tau}_{c,i} \in \mathcal{R}^p$ is the control vector input for subsystem i, $\boldsymbol{f}_i(\cdot) : \mathcal{R}^n \times \mathcal{R}^m \times \mathcal{R}^p \rightrightarrows \mathcal{R}^n$ is the vector of differential functions for subsystem i, $\boldsymbol{y}_i \in \mathcal{R}^r$ is the output vector and $\boldsymbol{h}_i(\cdot) : \mathcal{R}^n \times \mathcal{R}^m \times \mathcal{R}^p \rightrightarrows \mathcal{R}^r$ is the output mapping function vector for subsystem i.

Assume that these two subsystems are to be connected in a co-simulation such that $\boldsymbol{u}_1 = \boldsymbol{y}_2$ and $\boldsymbol{u}_2 = \boldsymbol{y}_1$. Then, $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are held constant between global time steps (macro time steps) T_d and updated only at each discrete time event $t_i \geqslant nT_d$ where $n \in \mathcal{N}_{\geqslant 0}$ is a counter that counts the number of data exchange events, starting at zero. The data exchange is controlled by a co-simulation master algorithm, as shown in Figure 2.4, which also controls the total simulation procedure and the global system time in the co-simulation.

Mathematically speaking, the two subsystems in the co-simulation end up being solved similar to the procedure given in Algorithm 2.1. Note that when multiple cores are used in the co-simulation (distributed simulation) the two **while**-loops given on the lines 7-12 and 13-18 in the algorithm are solved in parallel on their own cores, not in order for each other as shown in the algorithm. Also note that the sampling of data is done by the use of ZOH.

The procedure shown in Algorithm 2.1 is an explicit and non-iterative scheme to solve the two coupled subsystems. This, because the two subsystems can be connected to each other directly. However, there also exist iterative and implicit schemes as well [33]. While the explicit co-simulation procedure is by far the simplest and most straight forward one, the implicit co-simulation scheme gives

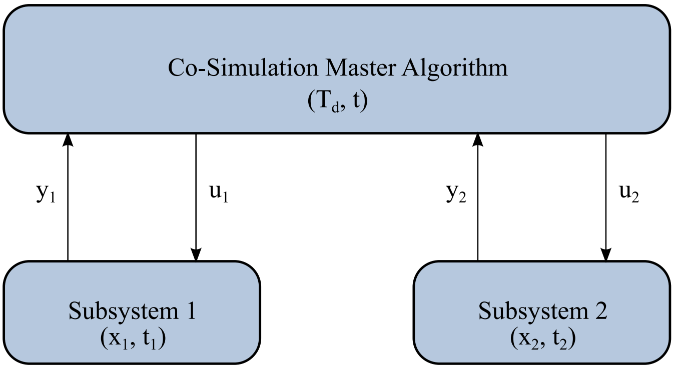


Figure 2.4: Two subsystems connected in a co-simulation

Algorithm 2.1 Solution procedure for a co-simulation containing two general subsystems.

```
procedure CoSim()                                    ▷ Total simulation implemented as a function
    Initialize()                                     ▷ Parameters and initial conditions
    while $t \leqslant t_{stop}$ do                  ▷ Solver loop
        $t = t + T_d$                                ▷ $T_d$ is the time between each communication points
        $\boldsymbol{u}_1 = \boldsymbol{y}_2$        ▷ Updating subsystem inputs
        $\boldsymbol{u}_2 = \boldsymbol{y}_1$
        while $t_1 < t$ do                           ▷ Solving subsystem 1 until next communication point
            $\dot{\boldsymbol{x}}_1 = \boldsymbol{f}_1(\boldsymbol{x}_1, \boldsymbol{u}_1, \boldsymbol{\tau}_{c,1}, t_1)$    ▷ Calculate rate
            $[\boldsymbol{x}_1,\ \Delta t_1] = \text{Solve}(\dot{\boldsymbol{x}}_1, \boldsymbol{x}_1, \boldsymbol{\tau}_{c,1}, t_1)$    ▷ Solve for next local time step
            $\boldsymbol{y}_1 = \boldsymbol{h}_1(\boldsymbol{x}_1, \boldsymbol{u}_1, \boldsymbol{\tau}_{c,1})$    ▷ Update subsystem output
            $t_1 = t_1 + \Delta t_1$                 ▷ Update local time
        end while
        while $t_2 < t$ do                           ▷ Solving subsystem 2 until next communication point
            $\dot{\boldsymbol{x}}_2 = \boldsymbol{f}_2(\boldsymbol{x}_2, \boldsymbol{u}_2, \boldsymbol{\tau}_{c,2}, t_2)$    ▷ Calculate rate
            $[\boldsymbol{x}_2,\ \Delta t_2] = \text{Solve}(\dot{\boldsymbol{x}}_2, \boldsymbol{x}_2, \boldsymbol{u}_2, \boldsymbol{\tau}_{c,2}, t_2)$    ▷ Solve for next local time step
            $\boldsymbol{y}_2 = \boldsymbol{h}_2(\boldsymbol{x}_2, \boldsymbol{u}_2, \boldsymbol{\tau}_{c,2})$    ▷ Update subsystem output
            $t_2 = t_2 + \Delta t_2$                 ▷ Update local time
        end while
        Collect($t, \boldsymbol{x}_1, \boldsymbol{x}_2 ...$)    ▷ Storing results
    end while
    plot($t, \boldsymbol{x}_1, \boldsymbol{x}_2 ...$)    ▷ Post-processing, for example plotting data
end procedure
```

additional support when the connectivity of subsystems in a co-simulation fails. Such connectivity problems are discussed in more detail in Chapter 4. Hence, when using iterative and implicit schemes the time steps in a co-simulation can be performed both in parallel (Jacobi) and in serial (Gauss-Seidel) [58]. However, in this thesis the focus will be given to the explicit one with constant input approximation between global communication time steps.

Even though Algorithm 2.1 shows a simplified mathematical representation of a co-simulation, it does not reveal anything about which functions in the subsystems that are called by the co-simulation master and how the data are exchanged between the subsystems, other than the fact that it happens after global communication time step T_d. If the two subsystems in the algorithm were two mathematical systems implemented in the same programming language, run on the same operative system and on the same computer (and possibly the same processor), the connection between the two codes would be simple, and one could easily tailor a co-simulation master algorithm that handles the signal flow as well as running the total co-simulation, although not being a generic implementation. However, if the subsystems are to be implemented and connected in a generic way in a co-simulation, having focus on reuse of the subsystems, the situation changes. Then a standard for distributed co-simulation systems that specifies the architecture and the implementation is needed. Such standards are discussed in the following.

2.3 Standards for Distributed Co-Simulations

As of today, two standards for performing distributed co-simulations are dominant, probably the most prominent one being the IEEE standard *High Level Architecture* (HLA) [59] and the newest one being the de-facto standard *Functional Mock-up Interface* (FMI) [60]. These two standards are quite different despite being somewhat interoperable. It is actually possible to run a distributed simulation using the HLA standard as master algorithm while implementing the subsystems according to the FMI standard [61, 62, 63]. In this work co-simulations will be based only on the FMI standard, as argued for in [33] where also a comparing presentation of the two standards are given. Hence, only a short abstract of the headlines from [33] is paraphrased in the following.

2.3.1 High-Level Architecture

High-Level Architecture (HLA) was initially developed for the US Department of Defense by the Defence Modeling and Simulation Office for use in wargaming and training simulations, and is a standard which describes a general-purpose co-simulation architecture [64]. The first version of HLA was released in 1996 and became a IEEE standard (IEEE 1516) a few years later. The latest version of this IEEE standard is IEEE 1516-2010 and is commonly referred to as *HLA Evolved* [65]. The implementation of HLA may vary and there exists both commercial and free HLA implementations. Two similar architectures are the *Distributed Interactive Simulation* (DIS) [66] and the *Common Simulation Interface* (CSI) [67]. DIS is the precursor of HLA and is particularly made for military applications with focus on meeting the training needs, while CSI was developed by SINTEF Ocean for the purpose of maritime vessel simulations.

HLA is designed around its subsystems, or subsimulators, called *federations*. A federation is a group of independent subsystems, known as *federates*, which communicate with each other through a common *Run-Time Infrastructure* (RTI), which is responsible for time synchronization as well as routing signals between the federates. The federates themselves can consist of various functionalities, such as for performing numerical simulations, enabling hardware and human interfaces and live visualization tools for training purposes or data monitoring.

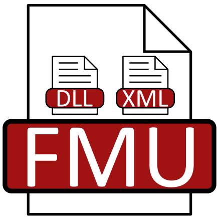
Some of the most commonly mentioned advantages of HLA include *interoperability*, which means that federates may run on different platforms and use different simulation methods, and *re-usability*, which means that federates may be easily re-used in multiple simulations. On the other hand, the HLA has also a few disadvantages. Because the wire protocol between the federates and the RTI is not standardized, a federate made for one specific HLA implementation generally can not be used with a different implementation. Also, in contrast to the master/slave structure in the FMI standard, federates are not passive slaves that wait for anonymous input data. Instead they actively request, by name, the data they require. This, in addition to other reasons such as not being sufficiently easy to use and not provide well-defined interfaces, is why the HLA standard was not selected as the co-simulation standard in the ViProMa project [33].

2.3.2 Functional Mock-up Interface

According to [60] the Functional Mock-up Interface (FMI) is a tool independent standard for the exchange of dynamic models and for co-simulations developed in the ITEA2 project MODELISAR in 2010. The standard is actively maintained and developed by the Modelica Association, and a second version was released in 2014 [68].

The standard specifies that each model should be packaged as a *Functional Mock-up Unit*, which are archive files that contain the model code for one or more platforms, along with metadata and model documentation. For example, for the Windows operating system a FMU would consist of shared libraries (dll) and a xml-file for representing the metadata, in additional to other supporting files. The standard also specifies the format and structure of files and directories in an FMU (see Figure 2.5), as well as the APIs that must be implemented by the model code, defined in the terms of the C programming language.

The FMI standard is divided into two parts, *FMI for Model Exchange* and *FMI for Co-Simulation*, where the latter is relevant here and defines an interface for models which are bundled with their own numerical solver. Hence, an FMU made based on this standard is a description of a subsimulator in the form of compiled



```
<fmu_name>
├ binaries
│ ├ win32
│ │ ├ <model_library>.dll
│ │ ├ <model_library>.lib
│ │ └ ...
│ ├ win64
│ │ └ ...
│ ├ linux32
│ │ └ ...
│ ├ linux64
│ │ └ ...
│ └ ...
├ documentation
│ ├ _main.html
│ └ ...
├ sources
│ └ ...
├ resources
│ ├ <dependent_libraries>
│ └ ...
├ modelDescription.xml
└ model.png
```

Figure 2.5: FMU structure

code and metadata. Several connected FMUs constitute a co-simulation, where a co-simulation master algorithm handles the data exchange between the FMUs and the time synchronization. As mentioned in section 2.2, the exchange of data is a discrete event taking place at each communication point, between which each model is solved independently from the others by its own local solver. However, it should be noted that the APIs do not put restrictions on whether a local solver is implemented in each FMU or not, it just tells the FMU to perform a time step of given length. Hence, an FMU can be implemented without a solver if not needed, which is typically the case if the FMU is a communication portal to hardware or human interface in a co-simulation.

The FMI standard for co-simulation is based on a master/slave model of communication where each FMU is a slave and controlled by a master algorithm. Hence, the FMUs in a co-simulation have no information about which other FMUs are present in the simulation, nor about which FMUs they are connected to, other than the fact that they know their input values, because all exchanged data in the co-simulation go through the master algorithm. Another interesting aspect regarding FMUs is that they are closed for modification, meaning that once an FMU has been created, there is not simple way of modifying its behaviour or external interface. This, because the model code is typically stored in compiled binary code form, which means that numbers, types and names of input and output variables are fixed. Even though this is bad for scalability and for making model changes on the fly, it enables sensitive model data to be exported as an FMU without spoiling business secrets, and is referred to as *black-box implementations*.

The FMI standard for co-simulation does not say how, or in what format, data are exchanged between the FMUs, nor how the the slaves are time synchronized. Hence, FMI support can be implemented in almost any type of simulation software and the number of tools supporting this standard is large and grows quickly [69].

2.4 Carrying out Co-Simulations

Even though the co-simulation standards provide a framework for implementing the subsystems in a co-simulation, it is not always trivial to determine the interface of each subsystem in a co-simulation such that each subsystem can be connected directly as intended. For sure one should try to implement each subsystem in a co-simulation with focus on modularity such that each subsystem can be replaced by another when testing different system configurations in design case studies. This is simpler when the same person has developed all subsystems in a co-simulation but this is seldom the case if co-simulations are to be used in the maritime industry. Even though system modularity and subsystem interfaces are discussed in more detail for a special type of systems in Chapter 4 a short discussion relating co-simulation scenario control and modularity will be presented shortly. Also, a short presentation of the vessel simulator derived in this thesis will be given in order to facilitate the discussion about co-simulation scenarios even though being thoroughly elaborated in section 8.3.

In this thesis the presented co-simulation case studies in Chapter 8 are simulated using the co-simulation master algorithm Coral developed in the ViProMa project, as will be presented in more detail in section 2.4.1, where each subsystem is implemented as FMUs according to the FMI standard. However, some smaller co-simulation case studies are also presented in Chapter 3 and Chapter 4, but these have been implemented directly in the Python programming language where each subsystem is implemented as function libraries and solved separately according to co-simulation theory. The co-simulation master algorithm is also implemented directly in Python and is similar to the one presented in Algorithm 2.1. Nevertheless, a short presentation of the co-simulation master algorithm Coral is given in the following.

2.4.1 Coral

The co-simulation software *Coral*, developed in the ViProMa project by SINTEF Ocean (former SINTEF Fisheries and Aquaculture), is built from the ground up with FMI support and has the same master/slave structure as the FMI standard. The software has two primary responsibilities, *communication* and *synchronization*. It transport data between the slaves according to the prior co-simulation connections, possibly over network, making sure that each output value is routed to the correct

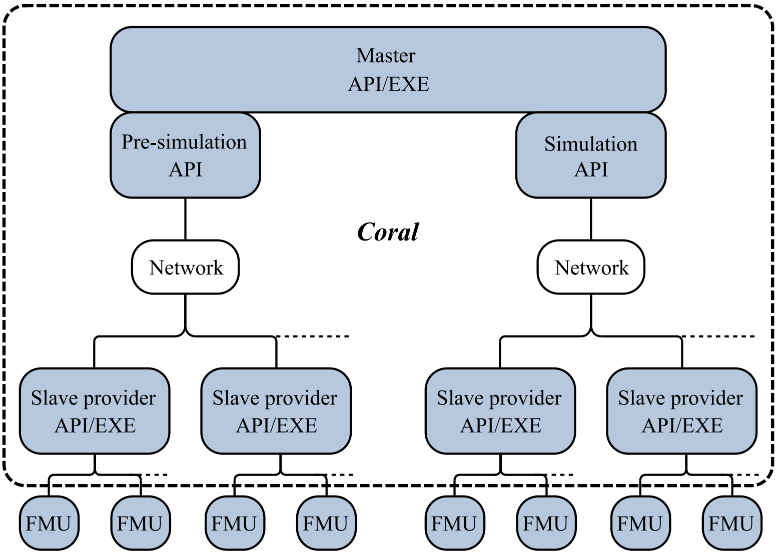


Figure 2.6: A diagram that shows the various components in a co-simulation when using the Coral software. Note that everything inside the dashed rectangle is formally part of Coral [33]. Note that the meaning of API/EXE in the figure is that the functionality is offered both in the form of a C++ programming interface and as a ready-made executable application.

input variables. The software also takes care synchronizing the slaves, meaning that it issues commands to all slaves that tell them when to perform a new time step and how large the time step should be before reaching the next communication point.

Since Coral supports network distribution of simulations, it is necessary to actually start the slaves on each computer when initiating a new simulation. This is done by a server program called *slave provider* which is started on each participating computer. This server program loads the FMUs that are locally available on each of the computers, publishes information about the available slaves on the network as well as spawning the slaves at the request of the co-simulation master. Note that an FMU is similar to a class in object-oriented programming terminology, which means that it represents a blueprint for a model and can be instantiated several times in a single co-simulation, representing multiple instances of that model in the simulation. A schematic overview of the master algorithm is shown in Figure 2.6.

Coral is implemented as a software library in C++ and can therefore be embedded directly into programs that need to perform co-simulations. In addition, it comes

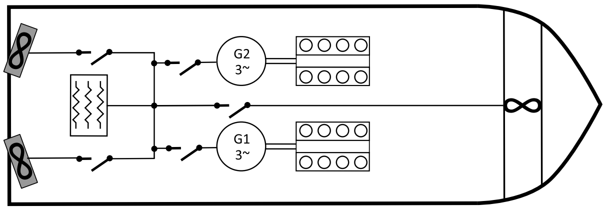


Figure 2.7: System overview of vessel in DP-operation including power plant and thruster configuration

with a set of stand-alone command-line applications that allow users to run co-simulations configured via text files.. The software is planned to be developed further in the future having additional useful functionalities and features, and has been released under a permissive open-source license both as pre-compiled and open code. A development schedule, documentation of the software, as well as a link to the download page of the software, can be found on the project's web-page [19].

One co-simulation system that has been simulated by the use of Coral is the open source vessel simulator derived in this thesis and a is presented in the following.

2.4.2 Vessel Simulator

Officially, two demonstrators have been developed in the ViProMa project, one being provided by the VISTA project [23] that uses the Coral software embedded in a workbench software developed in the project and the other one being a co-simulation of NTNU's research vessel R/V Gunnerus [70]. Since both these demonstrators are bound by licences; the VISTA project being an industrial project where the results are owned by VARD and the Gunnerus demonstrator containing submodels being restricted by license from VeSIM, a simulation software developed by SINTEF Ocean, an additional demonstrator has been developed in this thesis, not being restricted by any form of licenses. Most of the subsystems that constitute this demonstrator will be elaborated in Chapter 5, Chapter 6 and Chapter 7, and the application of the demonstrator will be presented in section 8.3. Nevertheless, a short introduction to the demonstrator itself is given here.

The demonstrator represents a generic offshore vessel in dynamic positioning (DP) operations including two azimuth thrusters placed at the stern as well as one tunnel thruster in the bow, a power plant, a wave filter for filtering oscillatory environmental disturbances, a DP-control system, a reference signal system and a thrust allocation algorithm, as shown in Figure 2.7. The subsystems constituting the vessel simulator is connected in a co-simulation environment as shown in Figure

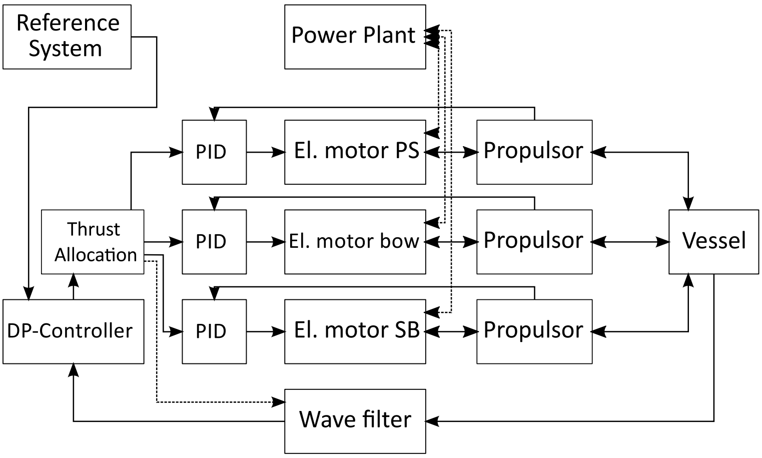


Figure 2.8: Overview of subsystems in the vessel simulator. Note that some of the connections are drawn with dashed lines in order to distinguish the connections from each other when they cross in the figure

2.8. Note that each thruster is respresented by a propulsor drive and a propulsor and that there are 15 submodels in the total co-simulation setup, in addition to monitoring FMUs for live plotting of data during a simulation that have not been included in the figure. These live monitors, along with some of the control systems have been implemented in a programming framework made by the author in C++ for fast prototyping- and automatic compilation of code and packaging of FMUs. Other models have been made in the 20-Sim modeling and simulation software [71], which has export tools supporting the FMI standard. Details regarding this demonstrator will be treated more thoroughly in Chapter 8.

In larger co-simulation systems, such as the one shown in Figure 2.8 system modularity and generic scenario handling is important if the co-simulation system is to facilitate replacing subsystems with others in a fast, efficient and robust manner. This is especially important if co-simulations are intended as a tool supporting the design process of new-builds, where some of the subsystems might be black-box models provided by third party vendors, as well as for rapid testing of different vessel configurations.

2.4.3 Simulation Scenario Control and Modularity

In co-simulations, system modularity, simulation scenario handling and configuration of co-simulations are all related to each other. Whether a system is modular or not is mainly dependent on the scope of study. For example, a marine offshore vessel implemented as one single subsystem including all its equipment in a co-

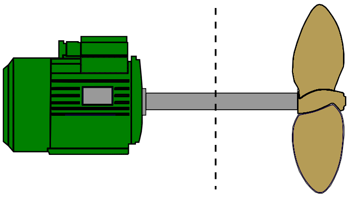


Figure 2.9: Sketch of an electrical motor powering a propeller. The dashed line in the figure represents a connection between the two systems in a co-simulation environment

simulation can be modular in the sense that it can be replaced by another total vessel model. However, it is not modular when for example considering replacing the propulsion system inside the subsystem. Then the changes need to be implemented in the subsystem and a recompilation is needed.

System modularity is also dependent on connectivity between subsystems and can be degraded if subsystems can not be directly connected to each other when configuring a co-simulations. In many cases it can be hard to determine which inputs and outputs to implement in a system, not only because of the causality of the system. In particular, when working with dynamical systems it can be difficult to determine which dynamical effects to include in which subsystem, especially when splitting one system into two. It is always preferred to obtain complete state-space representations of the systems, for various reasons discussed in the following chapters, but when one system is split into two one must determine how to split the dynamical effects between the two systems. For example, consider a propulsion unit consisting of a propulsor drive and a propulsor, connected in a co-simulation as in Figure 2.8. A closer view of the propulsion unit is shown in Figure 2.9, including an electrical motor and a propeller.

In the figure the vertical dashed line illustrates a possible way of splitting the propulsion unit into two subsystems – a propulsion drive and a propulsor. In this thesis three different ways of defining the inputs and the outputs for the two subsystems in the figure are considered and is dependent on in which subsystem the shaft speed is calculated. In the case study presented in section 8.3 the shaft speed is calculated in the propulsor model, which means that no rotational inertia is included in the propulsor drive. In the case study presented in section 8.5 the propulsor models are provided from VeSim and require a shaft speed input. Hence, the inertia of the propulsion unit is implemented in the propulsion drive model. The third option is to implement the shaft as a separate subsystem including the rotational inertia of the total propulsion unit. This option is considered in the case study presented in section 8.2. Also, a fourth option do exist, namely to split the rotational inertia such that both the propulsor drive and the propulsor output a shaft speed, but then a flexible shaft model between the subsystem is needed if not considering iterative and implicit co-simulation schemes. However, this option is not considered here since such a shaft model would become quite stiff in comparison to the other models and give additional room for numerical errors and state offsets, and would require a lower global communication time step size for obtaining good simulation results.

This small example illustrates that it is not always trivial to define subsystem connections, the inputs and outputs of each subsystem, and the choice of subsystem connections can have a significant impact on the simulation results. One way to increase the possibility to connect different subsystems in a co-simulation is to implement the respective subsystems with multiple input and output options. This is discussed in more detail in Chapter 4.

System modularity and effective configuration of co-simulations are also directly related to generic handling of simulation scenarios since a simulation scenario defines the operation conditions and possibly the overall situation control in a co-simulation, such as e.g. providing reference signals for the position and orientation for a marine vessel in DP-operations. This also sets requirements to the modularity and the model fidelity of each subsystem which needs to have the possibility to adapt to the simulation scenario. The fidelity of each connected subsystem will not be given much focus here since the subsystem fidelity depend on the focus of the co-simulation study, but the fidelity can have a significant impact on both the subsystem connectivity and the interaction effects between the connected subsystems.

On the other hand, the discussion here will focus on controlling the simulation scenario in a co-simulation in a generic manner. One example from the maritime industry that illustrates such a co-simulation scenario is the environmental loads affecting a vessel and its equipment. Such environmental loads can be waves, currents and winds and should have the same characteristics in each affected subsystem. To illustrate this in more detail, consider a marine offshore vessel performing a subsea crane operation as illustrated in Figure 2.10. In this case both the hull, the propusion units, the wire connecting the payload to the crane and the payload should experience the same global environmental conditions due to waves and currents, dependent on their global positions and orientations, and if high fidelity models are considered also interaction forces between the subsystems, such as between the thrusters and the hull, should be accounted for, as done in the case study presented in section 8.5. In such a co-simulation case one could implement the environmental conditions in each subsystem but this would degrade the system modularity because when replacing one subsystem by another the environmental conditions must be re-implemented, or at least re-configured in the new subsystem.

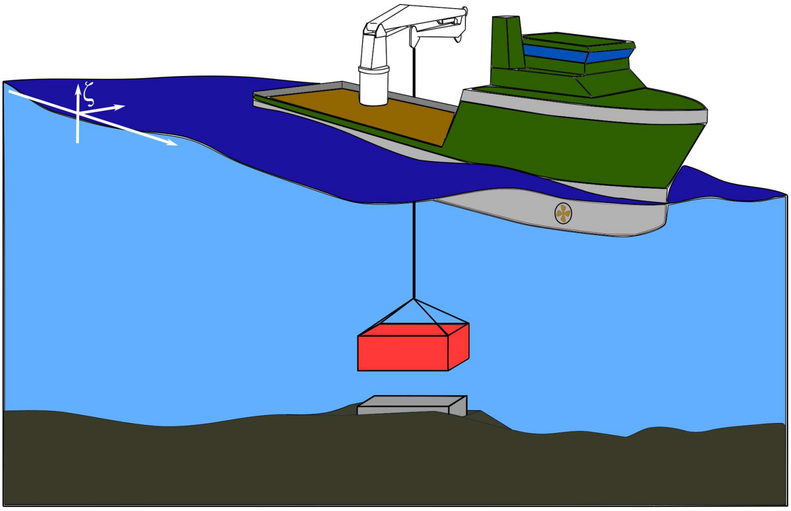


Figure 2.10: Sketch of an offshore marine vessel performing a subsea crane operation

Another problem with this approach is that the environmental conditions must be synchronized in each subsystem when interaction effects are considered. This argue for implementing the environmental conditions as a separate subsystem in a co-simulation where each subsystem that are affected by the environmental effects subscribes for the environment based on their global position and orientation. Then, a standardization of which data and how to exchange it between relevant subsystems should be considered.

As have been discussed in this section, the system modularity, system scenario control and configuration of co-simulations are strongly related. In this thesis specific subsystem connections and modularity will not be given much focus except for when the choice of subsystem inputs and outputs directly affects the state-space model by introducing differential algebraic equations, as will be discussed in Chapter 4. Also, it is out of scope in this thesis to develop a standard for handling scenarios in a generic fashion in co-simulations, mostly because of the importance of including the industry when developing such standards. Note that in the co-simulation case studies of marine offshore vessels presented in this thesis the environmental conditions will be implemented directly in the hull model, which other subsystems will subscribe to if necessary.

CHAPTER 3

Stability and Accuracy in Distributed Co-Simulations

This chapter is based on a reformatted and restructured version of [34, see **P2** in section 1.6], in addition to some topics that are presented in [35, see **P3** in section 1.6] and [36, see **P4** in section 1.6] as well as a general discussion about dynamical stability theories that are applicable to distributed systems. This chapter takes aim at studying numerical stability in distributed co-simulations through dynamical system stability and numerical stability using explicit numerical solvers. This is done by studying outer stability limits, for example stability conditions when handling unstable subsystems or marginally stable solvers. To conclude global stability of a distributed co-simulation both dynamical system stability and numerical stability must hold, and in this chapter these stability criteria is combined into one unified non-conservative criterion for distributed linear dynamical systems. Some examples are given in order to both highlight numerical stability issues and to prove stability in different case studies. The derived stability criterion is also extended to include distributed systems containing non-linear dynamics, although the stability results become more conservative. In the end, an adaptive non-iterative energy-conservation-based communication time-step size controller *ECCO* [35] is presented with the intention of improving the accuracy of the co-simulation results.

3.1 Introduction and Motivation

Stability is a widely used term in the field of modeling and is used both in context of dynamical systems and solutions in simulations. However, dynamical stability and numerical stability are often separated, even though both are somewhat dependent on the system characteristics, because they usually have separate areas of application. In general, stability in systems is a measure of convergence, often expressed through asymptotic- or exponential characteristics, and is solely related to dynamical system properties. Dynamical stability is well documented in the field of modeling [25, 31, 27], as well as in the field of control [30, 29]. On the other hand, numerical stability in solutions is a measure of numerical convergence [72] and is dependent on the eigenvalues of the system to be solved and the numerical solver characteristics. In other words, the stability of a system is a property characterized by the system dynamics and the stability of solutions is a solver property, whether the eigenvalues of the system are inside the stability region of the chosen numerical solver or not. However, in distributed co-simulation systems the stability of the system and the solution seems to be more related.

Dynamical stability of distributed systems is similar to sampled system stability [73, 74]. However, when using fixed-step size solvers, the dynamic stability of linear distributed systems is closely related to the stability of discretized linear systems [29, 75]. Stability of distributed systems has been studied from many different angles in the literature, for example through zero-stability analysis of coupled integration [76, 77], jacobian-based co-simulation algorithm to overcome stability issues [76], stability and convergence analysis of sequential algorithms [78], modular integration for Runge-Kutta methods [79] and Dahlquist test equations [80] for stability analysis of distributed systems [81, 82]. However, less results containing numerical stability of distributed simulation results can be found in the literature.

In this chapter numerical stability of distributed co-simulations will be studied on a general level, and much focus will be given to linear systems. Also, some focus is given to dynamical stability of distributed systems where linear systems and transfer functions as well as general passivity theory will be highlighted. It should be noted that a thorough stability analysis of a distributed simulation system is contradicted by the use of black-box models since all system information are hidden. As it turns out, some subsystem characteristics can help us guarantee stability without considering the detailed subsystem information. This topic is discussed in section 3.3.2. For the rest of this chapter, all subsystem information in a co-simulation are considered known. Before diving into the core topics in this chapter, some background and definitions are given in the following.

3.2 Background Theory and Definitions

In this chapter, the main focus is given to linear dynamical systems, since the explicit numerical solvers are linear and, as will be shown later on, some of the results from studying linear systems can be extended to also yield for non-linear systems, although being more conservative. In the following, a general linear system is defined and a presentation of explicit numerical solvers are given.

3.2.1 Linear System Formulation

If we assume that the output $\boldsymbol{y}_i$ for subsystem i is only dependent on the internal states in a subsystem, (2.6) may for subsystem i be rewritten as

$$\begin{aligned} \dot{\boldsymbol{x}}_{\boldsymbol{i}} &= \boldsymbol{A}_i \boldsymbol{x}_i + \boldsymbol{B}_i \boldsymbol{u}_i + \boldsymbol{B}_{c,i} \boldsymbol{\tau}_{c,i} \\ \boldsymbol{y}_i &= \boldsymbol{C}_i \boldsymbol{x}_i \end{aligned} \tag{3.1}$$

where $\boldsymbol{A}_i \in \mathcal{R}^{n \times n}$ is the state mapping matrix, $\boldsymbol{B}_i \in \mathcal{R}^{n \times m}$ is the input matrix mapping, $\boldsymbol{B}_{c,i} \in \mathcal{R}^{n \times p}$ is the control matrix mapping and $\boldsymbol{C}_i \in \mathcal{R}^{r \times n}$ is the output mapping matrix for a set of linear differential equations with n states, m inputs, r outputs and p control variables. If only two single uncontrolled subsystems are present in the distributed system, the differential equations can be simplified and expressed as

$$\begin{aligned} \dot{x}_1 &= a_1 x_1 + b_1 u_1 \\ y_1 &= c_1 x_1 \end{aligned} \tag{3.2a}$$

$$\begin{aligned} \dot{x}_2 &= a_2 x_2 + b_2 u_2 \\ y_2 &= c_2 x_2 \end{aligned} \tag{3.2b}$$

where y_1 and y_2 are the subsystem outputs, and at each communication point

$$\begin{aligned} u_1 &:= y_2 \\ u_2 &:= y_1 \end{aligned} \tag{3.3}$$

In the rest of this chapter it will be assumed that $\boldsymbol{\tau}_{c,i} = \boldsymbol{0}$ for simplification reasons. This can be argued for since the internal control law often can be represented as a state-space formulation as well, possibly increasing the size of $\boldsymbol{x}_i$. Before discussing dynamical and numerical stability of distributed systems a short introduction to explicit numerical solvers are given in the following.

3.2.2 Explicit Numerical Solvers

In this chapter only explicit local solvers are considered in each subsystem in a co-simulation, since only explicit co-simulation schemes are treated. Hence, no port-dependent algebraic loops and relations between connected subsystems in a co-simulation are considered. Such algebraic port-dependent loops and implicit equations are discussed in more detail in Chapter 4. A thorough introduction to numerical solvers are given in [83] and only a short introduction will be given here. For simplification reasons, the solvers used here are also assumed to have fixed time-step sizes. This is because the same conditions for assuring stable simulation results will yield for variable time-step sizes, where the maximal (and in some cases the minimal) local time-step sizes are determined.

The stability of a numerical solver is closely related to the eigenvalues in the system to be solved, and by knowing the eigenvalues it is possible to choose a solver and a time-step size that stabilize the solution. Moreover, such solvers are also linear which means that when solving non-linear differential equations the solvers approximate the solution of the system by solving piecewise linearized parts of the non-linear system.

For a system i, given as in (3.1), the eigenvalues λ_i can be found by solving

$$\det(\boldsymbol{I}\lambda - \boldsymbol{A}_i) = 0 \tag{3.4}$$

where $\boldsymbol{I}$ is the diagonal unit matrix of size $n \times n$ where n is the number of states in the subsystem. By assuming that the forward Euler integration method is used one can find the largest time step for which the solution is stable based on these eigenvalues, assuming that the system inputs $\boldsymbol{u}_i$ and $\boldsymbol{\tau}_{c,i}$ are held constant. A differential equation given as

$$\dot{x} = f(x) \tag{3.5}$$

can be solved by the forward Euler integration method as

$$x_{i+1} = x_i + f(x_i)\Delta t \tag{3.6}$$

where x_{i+1} is the numerical solution of the differential equation at time $t_{i+1} = t_i + \Delta t$. The forward Euler integration method is stable if $\forall\ \lambda_i$,

$$|1 + \lambda_i \Delta t| \leqslant 1, \quad \Delta t_i > 0 \tag{3.7}$$

where λ_i is eigenvalue $i \in \{1, ..., n\}$ of the differential system. If the eigenvalues are all real, the stability criteria can be simplified to

$$\Delta t \leqslant -\frac{2}{\lambda_i} \tag{3.8}$$

By looking at the Tailor series expansion of the exponential function given as

$$e^x = \sum_{j=0}^{\infty} \frac{x^j}{j!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + ... \tag{3.9}$$

we see that the Euler integration method approximates the exponential function with an order of 1. Hence, the Euler integration method is a first order method. The Euler integration method equal to the first order Runge-Kutta method, and the

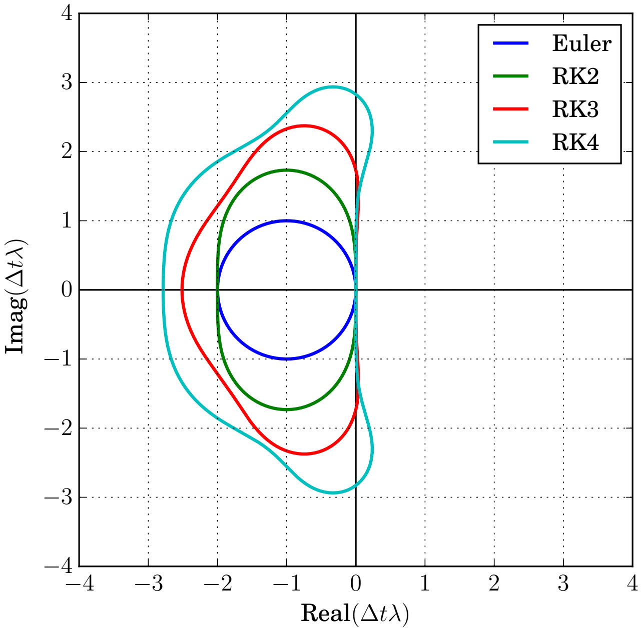


Figure 3.1: Stability regions for explicit numerical solvers. The eigenvalues of the system λ_i multiplied with the solver time step Δt_i must be placed within these limits for the respective solver to remain stable. Note that *RK* is the abbreviation for Runge-Kutta and the following number gives the order of the method

stability limit for higher order Runge-Kutta methods can be found similarly. For simplification reasons let the approximation of e^x with an order p be defined as

$$\tilde{Exp}(x,p) = \sum_{j=0}^{p} \frac{x^j}{j!} \tag{3.10}$$

Hence, the stability criterion for the Runge-Kutta 4 method can be expressed as

$$|\tilde{Exp}(\Delta t\lambda, 4)| = |1 + \Delta t\lambda + \frac{\Delta t^2\lambda^2}{2} + \frac{\Delta t^3\lambda^3}{6} + \frac{\Delta t^4\lambda^4}{24}| \leqslant 1. \tag{3.11}$$

Figure 3.1 shows a graphically representation of the stability limits for the forward Euler integration method and the Runge-Kutta integration method with orders 1 to 4. From the figure we can clearly see that higher order explicit numerical integration methods, such as the Runge-Kutta 4 integration method, allow larger time-step size than lower order methods such as the forward Euler integration method.

Hence, by proving that a system is stable for a given time-step size with the forward Euler integration method, we are guaranteed that it will remain stable, using the same time-step size, with higher order explicit integration methods.

Explicit numerical solvers have the property of being stable when solving explicit dynamical systems with stable dynamics, assuming the local time steps have been chosen correctly and that the total system has no inputs ($\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$) that can destabilize the system. This, because $|e^{\Delta t \lambda}| \leqslant 1$ implies that $\lambda \leqslant 0$. As it turns out, if the dynamics in a system are unstable, the solution of the system when using an explicit numerical solver would also become unstable. Before moving on to numerical stability for distributed co-simulation systems, a review of applicable dynamical stability theories is conducted and presented in the following, with a main focus on linear systems and their transfer functions, as well as general system passivity theory.

3.3 Dynamical Stability

Normally, linear systems and non-linear systems are separated when talking about dynamical stability. This is not because they are treated differently, because linear systems can be analysed the same way as non-linear systems. However, since linear system theory is a special case of non-linear system theory, there exists additional stability results, tools and methods for analysing linear systems, that are not directly applicable to non-linear systems, which result in less conservative stability results than for non-linear systems.

When analysing dynamical stability of a distributed system, all system information must be known if the subsystems do not have certain properties, as will be discussed in section 3.3.2. Before presenting dynamical stability theory related to linear systems and other special cases, a general discussion regarding different stability theories is given in the following.

3.3.1 General Stability Theory

When analysing the dynamical stability of continuous systems the most widespread theory is the Lyapunov stability theory. This stability theory is an energy-based method for assuring that the dynamics in a system dissipate enough energy to remain stable. In other words, if a system dissipate more energy than it produces, it will converge to the nearest equilibrium point [30, 84]. The difference in energy production and energy dissipation in a system is related to the eigenvalues in the system and if a system has an increase in energy dissipation while the energy production remains constant, the eigenvalues in the system becomes more negative, as will typically the rate of the Lyapunov function. Note that *energy* is a loose term in Lyapunov stability theory and does not necessarily reflect the actual energy in the system and is more a measure of state values which typically reflects the energy in a system. Lyapunov stability theory has its major strengths when it comes to control system designs, and many control designs such as *backstepping* [85] and *sliding mode control* [86] can be derived directly from such a stability analysis [87, 28].

When it comes to sampled systems, or distributed co-simulation systems, Lyapunov stability theory can be applied as well [88]. However, a co-simulation system must be represented in a suited way, mathematically speaking. There exists many different frameworks for describing a sampled system mathematically. Three of these frameworks include a complete discrete-time system representation, which means that non-linear dynamics must be approximated, a more or less complete continuous-time system but where sampled inputs are treated as arbitrarily large constants for assuring input-to-output stability [89], and represented as a combination of a continuous-time and discrete-time system, as is done in *hybrid dynamical systems* [90, 91].

When representing a sampled system as a discrete-time system, non-linear dynamics are approximated as linear dynamics, for example by using the Euler discretization method [92]. The rate criterion for the Lyapunov function is also split into two, one criterion for continuous dynamics as before, and one rate criterion for assuring stability in discrete events, such as when exchanging data between subsystems global communication time steps in co-simulations. A thorough review of discrete Lyapunov stability theory is given in [93], and some interesting aspects regarding non-linear sampled systems are discussed in [94]. Even though non-linear dynamics are approximated in discrete-time system representations, the system can be proven stable based on the approximated systems if having certain characteristics [95]. This is also proven for the hybrid dynamical systems framework [96].

In general, these three methods for representing a co-simulation system mathematically fit all dynamical systems represented as differential equations, both linear and non-linear. However, in practice, the introduced dynamical stability frameworks may not be as simple to apply in practice, since all information about the subsystems must be known and since suited Lyapunov function candidates must be constructed. These Lyapunov function candidates may be harder to construct for sampled systems, at least in the hybrid dynamical framework since it must satisfy both the continuous-time and the discrete-time rate criteria. In [90, Example 3.21] an example describing the dynamical stability analysis of a linear sampled-data system using Lyapunov stability theory is given, but the example is restricted to systems that have special passivity characteristics.

There exists a few stability theories that enable us to analyse each subsystem in a co-simulation separately and sum each local stability results into a global system stability result. Two such stability theories are *Input-to-State-Stability (ISS)* [97] and system passivity theory. Passivity is a special and quite useful dynamical system property and can save us a lot of work when analysing the stability of a distributed system.

3.3.2 Passivity

Passivity is an input-output property of dynamical systems that reflects the dissipative properties in a system [98]. Passivity is also closely related to dynamical stability [99, 100], such as the finite gain $\mathcal{L}_2$ stability [101]. Moreover, if multiple passive systems are connected, the total connected system is also passive [102], which yields both for continuous-time systems and discrete-time systems. This means that one can assure stability of a co-simulation including black-box models as long as the provider of the black-box models can assure that the models have certain passivity characteristics. In other words, we do not need to know the dynamics in a black-box model in order to assure that our total co-simulation is dynamically stable as long as the black-box model has certain passivity characteristics.

Passivity theory has been thoroughly documented for continuous-time systems, and good introductions to passivity are given in [101, 103, 104]. Some work regarding passivity in discrete-time systems are also documented in the literature. Several dissipativity and passivity implications for linear discrete-time systems are discussed in [102], feedback passivity of non-linear discrete-time systems with a direct link to the system input and output is discussed in [105], passivity with respect to discrete-time lossless systems are discussed in [106, 107] and the concept of average passivity is introduced for differential difference presentation of non-linear discrete time dynamics in [108]. However, we do not need to apply discrete-time passivity theory to co-simulation systems, since each subsystem in the co-simulation is assumed continuous and were only subsystem inputs and outputs are characterized as discrete. As a matter of fact, if all subsystems in a co-simulation have the right passivity properties, the total co-simulation system is not only stable but we might as well choose the global communication time step T_d arbitrarily.

The passivity theory provides multiple useful stability theorems and two of the most central ones are given in [101, Theorem 6.2 and 6.3]. For example, by using Theorem 6.2 in [101] it is possible to prove that the two subsystems in a co-simulation given as

$$\begin{aligned}\dot{x}_1 &= -x_1 + 0.1u_1 \\ y_1 &= x_1 \\ \\ \dot{x}_2 &= -0.5x_2 + 0.1u_2 \\ y_1 &= x_2\end{aligned} \tag{3.12}$$

are finite gain $\mathcal{L}_2$ stable. Moreover, it can be proven that the two systems are strictly passive, which according to Theorem 6.3 in [101] results in global asymptotically stability if Lyapunov functions given as $V_i = \frac{1}{2}x_i^2$ are used. Hence, for the co-simulation system given in 3.12 we can choose T_d arbitrary without affecting the dynamical stability of the co-simulation system.

Another special case in dynamical stability analysis is linear systems where stability analysis methods utilizing transfer functions can be applied. This method is exact for linear systems but can also be used to assure stability of non-linear systems by using linear approximation functions.

3.3.3 Linear Systems and Transfer Functions

Perhaps one of the major advantages of working with linear sampled systems is that the systems themselves can be exactly represented by transfer functions. Since we already have a transfer function describing the ZOH, see (2.5), we can simply combine the subsystems and the sampling dynamics in the stability analysis [29]. Let us assume that we have two singe-input-single-output (SISO) linear subsystems as given in (3.2) which are connected in a co-simulation and that exchange data with the communication time step T_d. The dynamics in each subsystem can be represented as a transfer function given as

$$H_i(s) = \frac{y_i(s)}{u_i(s)} = \frac{b_i c_i}{s - a_i} \tag{3.13}$$

By combining the system dynamics with the sampling dynamics in (2.5) the total transfer function can be expressed as

$$H_i(s)H_{ZOH}(s) = \frac{b_i c_i}{s - a_i}\left(\frac{1 - e^{-sT_d}}{s}\right) \tag{3.14}$$

A block diagram representation of the total system is shown in Figure 3.2. Note that two noise variables v_1 and v_2 has been introduced in order to calculate a total system transfer function that can be used in a stability analysis. For example, by setting $v_2 = 0$ one can calculate a transfer function that relates the noise v_1 to the

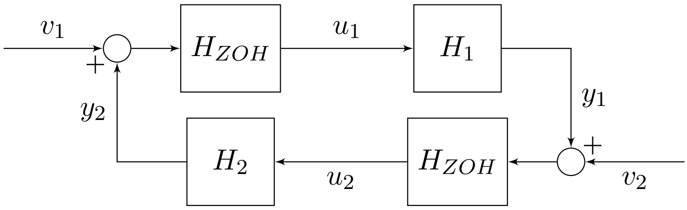


Figure 3.2: Block diagram representing a co-simulation including two linear SISO subsystems and zero-order-hold sampling blocks

output y_1. If this transfer function is stable, then the total co-simulation system would also be stable. In order to simplify the calculations, the transfer function given in (3.14) is transformed into the $\mathcal{Z}$-plane and expressed as

$$\mathcal{Z}\left[H_i(s)H_{ZOH}(s)\right] = \frac{b_i c_i}{a_i}\left(\frac{e^{a_i T_d}-1}{z-e^{a_i T_d}}\right) \tag{3.15}$$

where z is the z-transform operator. Hence, the transfer function relating v_1 to y_1 can after some calculations be expressed as

$$H_{v_1 y_1}(z) = \frac{y_1(z)}{v_1(z)} = \frac{\sigma_1(z-e^{a_2 T_d})(e^{a_1 T_d}-1)}{(z-e^{a_1 T_d})(z-e^{a_2 T_d})-\sigma_1\sigma_2(e^{a_1 T_d}-1)(e^{a_2 T_d}-1)} \tag{3.16}$$

where $\sigma_i = \frac{b_i c_i}{a_i}$. To evaluate weather the transfer function is stable or not, we must find the poles of the transfer function which is done by finding for which values of z the denumerator in the transfer function is zero. If these values of z are less than or equal to 1 in magnitude, the total system is stable. Hence, the transfer function in (3.16) is stable if

$$\frac{1}{2}\Big|e^{a_1 T_d} + e^{a_2 T_d} \\ \pm\sqrt{(e^{a_1 T_d}+e^{a_2 T_d})^2 + 4\sigma_1\sigma_2(e^{a_1 T_d}-1)(e^{a_2 T_d}-1) + 4e^{(a_1+a_2)T_d}}\Big| \leqslant 1 \tag{3.17}$$

It is hard to solve (3.17) explicitly for T_d, but one can solve it graphically by plotting the stability criteria as functions of T_d. An example that illustrate this method is given in 3.1 .

Example 3.1 (Stability analysis using (3.17)). *Assume that two connected subsys-*

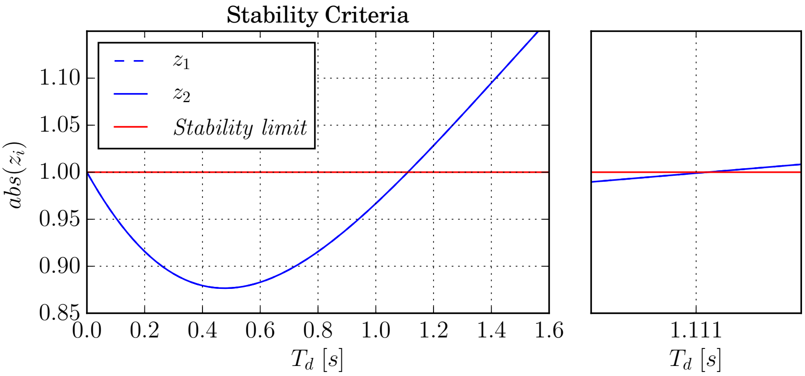


Figure 3.3: Stability criteria for transfer function in (3.16)

tems are given as

$$\begin{aligned}\dot{x}_1 &= -x_1 + u_1 \\ y_1 &= x_1 \\ \\ \dot{x}_2 &= -0.1x_2 - u_2 \\ y_2 &= x_2\end{aligned} \tag{3.18}$$

Figure 3.3 shows the real values of the poles z_i for the transfer function in (3.16). *As can be seen in the figure, the two poles seem to overlap and the system is stable as long as 0 s$\leqslant T_d \leqslant$1.111 s. This proves that we have found an upper limit for T_d for which the system is stable when a constant noise bias v_1 is added to the input of subsystem 1. Since the entire system is represented in the transfer function, it also proves that we have found an upper limit for T_d for which the entire system is stable even though $v_1 = 0$.*

Figure 3.4 shows a co-simulation of the two connected subsystems where $T_d =$ 1.111 s, the initial conditions are set to $x_1(0) = 5.0$ and $x_2(0) = 2.0$, and where the Euler integration method is used as a local numerical solver in both subsystem with a time-step size of 0.001 s. As can be seen in the figure, both states seem to be marginally stable, meaning that T_d can not be increased any further without making the entire co-simulation unstable. Note that Global solution in the figure refers to the sampled data while Local solution refers to the output from each of the local solvers. ◇

In [109] a stability criterion for sampled-data linear systems derived from Lyapunov functions and the hybrid dynamical framework, resulting in a set of linear

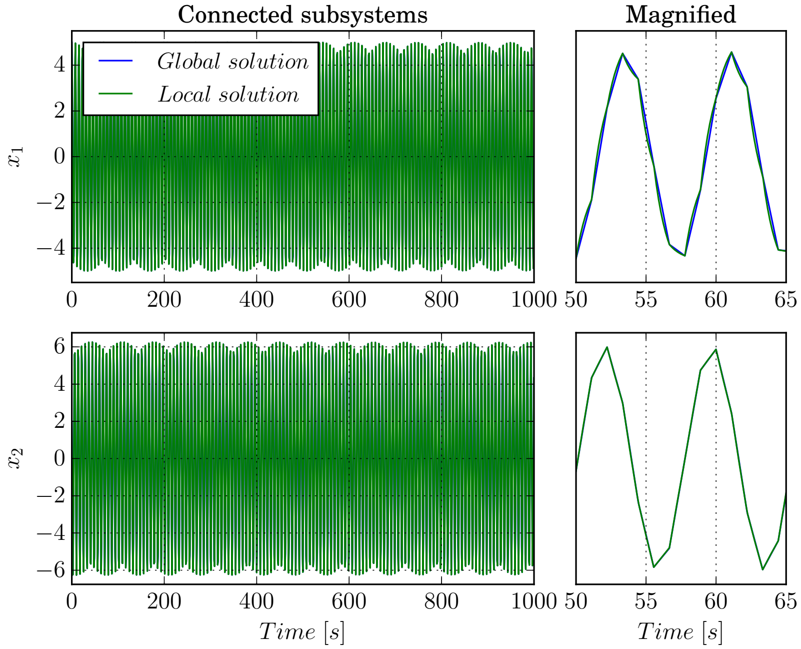


Figure 3.4: Simulation of connected subsystems, $T_d = 1.111s$, $x_1(0) = 5.0$ and $x_2(0) = 2.0$. The Euler integration method is used locally to solve both systems with a time step of $0.001\ s$ ($\frac{1}{\Delta t_i}$ where Δt_i is the local solver time step for subsystem i), and the green graphs denote the local solutions for the Euler solver plotted with a frequency of $1000\ Hz$ while the blue graphs denote the global solution plotted with a frequency of about $0.9\ Hz$ ($\frac{1}{T_d}$)

matrix inequalities (LMIs) that needs to be evaluated, is presented. By applying this criterion to our system in (3.18) we obtain a maximal stabilizing value for the global communication time step of $T_d = 0.9415\ s$, which is a more conservative result.

In the co-simulation in Example 3.1 the local solver time-step sizes were set relatively low in comparison to the eigenvalues of the systems. This was done in order to minimize numerical errors due to the solvers when studying the stability limit for the two connected systems. However, the two systems could have been solved separately with much higher solver time-step sizes, as will be discussed in 3.4.1, but since the co-simulation is marginally stable when $T_d = 1.111\,\mathrm{s}$, an increase in both local solver time-step sizes will decrease the system robustness and may result in an unstable simulation. Another stability aspect is that the local propagating times in each subsystem in a co-simulation must be synchronized at each global communication point, as will be elaborated in section 3.4.2. Hence, the choice

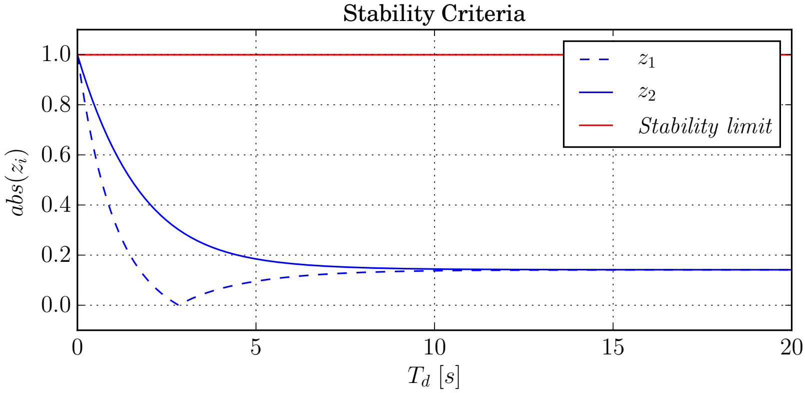


Figure 3.5: Stability criteria for the passive co-simulation system presented in (3.12)

of T_d also affects the choice of local solver time steps. Nevertheless, by setting the local time steps in the linear co-simulation system in Figure 3.4 to 0.00275 s, while T_d is left unchanged, the total co-simulation becomes unstable.

Now, when we have an exact dynamical stability criterion for two linear subsystems connected in a co-simulation, we can take another look at the passive linear co-simulation system presented in (3.12).

Example 3.2 (Stability criterion applied to passive subsystems)**.** *Figure 3.5 shows a plot of the eigenvalues for the total system given in* (3.12) *as a function of T_d in comparison to the dynamical stability limit. As can be seen in the figure, the eigenvalues stabilize at $|z_i| = 0.1414$ when $T_d \rightarrow \infty$. This can also be proven mathematically by setting $T_d \rightarrow \infty$ in* (3.17), $\lim_{T_d \rightarrow \infty} |z_i| = | \pm \sqrt{\sigma_1 \sigma_2}| = \sqrt{0.02}$. *Hence, the system is stable for $T_d > 0$. Note that since each local solver time step in a co-simulation must be less than or equal to T_d, the global communication step must be larger than zero in order for the co-simulation to propagate in time.* ◇

It is clear that it is a lot of work to analyse the stability of a large co-simulation system, even though only consisting of linear subsystems where system representations through transfer functions are possible. Also, even though a co-simulation is proven stable for a given value of T_d, it might not be stable in a simulation. This has to do with the numerical stability of the co-simulation itself. As was established in section 3.2.2 a given value of T_d affects the eigenvalues of the co-simulation system, and hence, the local numerical solvers. Thus, a co-simulation system must be both dynamical stable and numerical stable in order to produce stable simulation results as will be discussed in the following.

3.4 Numerical Stability in Distributed Systems

When studying a single dynamical system one can determine which numerical solver to use and the corresponding solver time-step size based on the system dynamics if the numerical solver is an explicit fixed-step size solver. However, in a distributed system the inputs in each subsystem will affect the total distributed system dynamics and change the eigenvalues respectively. This can cause instabilities in the local solvers and will be discussed more in detail in the following.

3.4.1 Continuous System Analysis and Eigenvalues

A single linear differential subsystem given as in (3.2a) has the eigenvalue $\lambda_1 = \frac{1}{a_1}$ and it can be verified that if $a_1 < 0$ and that u_1 is bounded and equal to some constant, the system is dynamically stable when assumed continuous. By assuming that the Euler integration method is used to solve this subsystem, the time step requirement for a stable solution is given as $\Delta t_1 < \frac{-2}{a_1}$.

Now, consider two single uncontrolled linear differential equations as given in (3.2) with $u_i = y_k \; \forall i \neq k$ where k is the last communication time step. These differential equations can be rewritten as a compact set of differential equations in continuous time under the assumption that the two subsystems interchange data continuously,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} a_1 & b_1 c_2 \\ b_2 c_1 & a_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.19}$$

and in an even more compact form

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} \tag{3.20}$$

where $\boldsymbol{x} = [x_1, x_2]^\top$. This system will always be dynamically stable when treated as a continuous system of differential equations if all eigenvalues of $\boldsymbol{A}$ are less than zero. The eigenvalues in (3.19) can be calculated as

$$\lambda_{1,2} = \frac{1}{2}\left[a_1 + a_2 \pm \sqrt{(a_1 - a_2)^2 + 4b_1 b_2 c_1 c_2}\right] \tag{3.21}$$

As can be seen when treating the two subsystems as one continuous system, the eigenvalues for the total system are different from the eigenvalues in each subsystem, as long as $\{b_1, b_2, c_1, c_2\} \neq 0$. Hence, when two or more subsystems with local numerical solvers are connected in a distribute system the stability of the local solvers would also depend on the interacting dynamics between the subsystems. This can be a problem when using fixed step solvers such as the forward Euler integration method if a predefined time-step size for each subsystem is be set without knowing the dynamics of the surrounding subsystems in a distributed simulation.

One could argue that by choosing the local solver time-steps as small as possible the simulation results from the total system would also become stable. However, it is not certain that the combined dynamics are stable, and even if they are, the time it takes to solve the total system would increase significantly. It is then reasonable to believe that by decreasing the communication frequency in a distributed system, the interconnected dynamics get weakened and the total simulation results in the distributed system may remain stable. This is summarized in Lemma 3.1.

Lemma 3.1. *A set of connected subsystems in a distributed system that are dynamically stable by themselves, having their own local fixed-step size solvers with large time-steps that keep the unconnected subsystems stable by themselves, may not result in stable solutions in a distributed setting due to the interconnected dynamics when the global distributed time-step goes size to zero, but will become stable when the global time-step is set large enough.*

To be able to stabilize the local solvers, one must find a global communication time-step size T_d that reduces the effects of the interconnected dynamics such that the local numerical solvers remain stable. However, it is believed that there may be some restrictions related to sampling and signal processing theory when choosing the global communication time-step size. These topics will be studied in the following.

3.4.2 Combined Distributed System Stability

The example and the discussion presented in the previous section, roughly summarized in Lemma 3.1, give a good introduction to issues regarding distributed simulation stability. However before starting with any stability analyses, a few thoughts and comments regarding expected restrictions need to be mentioned. To be more specific, Lemma 3.1 points to a solution of the stability problem where the phenomena of *aliasing* is utilized. Aliasing in sampling theory is when the sampling frequency is chosen too low such that the sampled data fails to represent the sampled system as illustrated in Figure 3.6. This means that we want to set T_d large, such that the communication frequency becomes low enough for the local solvers to become stable. However, aliasing often introduces distortions and numerical errors and will be given more attention to later on. The question that now comes to mind is why we in the first place want to utilize aliasing and fail to represent the system interactions in a proper manner. The answer is however rather simple. If we do not have the opportunity to choose the local solver time-step sizes in the subsystems, there may exist system configurations where the solution of the total system becomes unstable. We then have two options. We can choose to not use the system in simulations because we are not allowed to change the local time-step sizes, or, if we are not interested in transient simulation results, we may

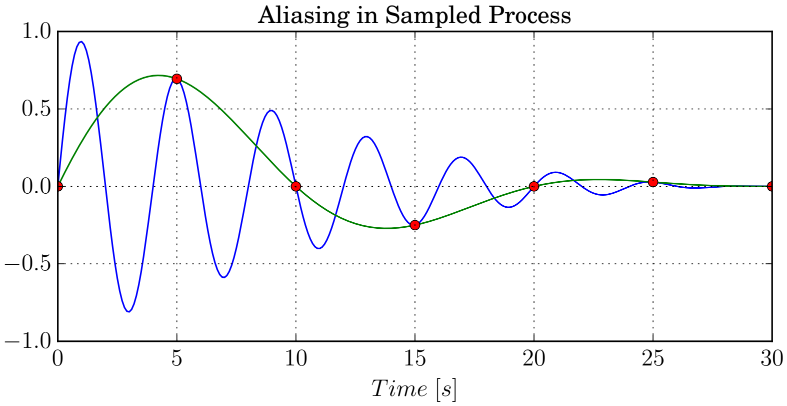


Figure 3.6: Aliasing in sampled system. The blue graph represents the process to be sampled, the read dots are the sampled values and the green graph is the recreated process from the sampled data.

choose T_d such that we at least are able to obtain a correct and stable solution after an incorrect transient simulation period. In many cases the latter is the preferred option.

Before moving on with the analysis we need to define the concepts *local* and *global solutions*. A local solution is here defined as the solution obtained locally in one connected subsystem, in comparison to a global solution, which is the collection of local solutions sampled with T_d. In this study the stability of both these solutions would be of interests since they strongly depend on each other, meaning that if all local solutions are stable with a given T_d, so is the global solution. However we can not guarantee that the local solutions are stable based on a stable global solution due to the aliasing side effects.

Assume that a single linear differential equation is given as in (3.2a), representing subsystem i in a distributed simulation, and is to be solved locally with the forward Euler integration method, as in (3.6). Hence, the solution of the subsystem can be calculated as

$$x_i(k+1) = x_i(k) + \Delta t_i \left(a_i x_i(k) + b_i u_i(k)\right) \tag{3.22}$$

for a given time step k. By defining that

$$n_i := \frac{T_d}{\Delta t_i}, \tag{3.23}$$

is the number of local solver time steps before receiving a new input and if k was the last communication point,

$$u_i(k) = u_i(k+1) = ... = u_i(k+n_i) \tag{3.24}$$

Since subsystem i is not communicating with the rest of the distributed system in the time interval $t \in [k, k + n_i]$, the system only depends on itself and the last input value $u_i(k)$. Hence,

$$\begin{aligned}
x_i(k+n_i) &= x_i(k+n_i-1) \\
&+ \Delta t_i \left(a_i x_i(k+n_i-1) + b_i u_i(k)\right) \\
&= (1+\Delta t_i a_i) x_i(k+n_i-1) + \Delta t_i b_i u_i(k) \\
&= (1+\Delta t_i a_i)^2 x_i(k+n_i-2) \\
&+ \left[(1+\Delta t_i a_i) + 1\right] \Delta t_i b_i u_i(k) \\
&= (1+\Delta t_i a_i)^3 x_i(k+n_i-3) \\
&+ \left[(1+\Delta t_i a_i)^2 + (1+\Delta t_i a_i) + 1\right] \Delta t_i b_i u_i(k) \\
&= \dots \\
&= (1+\Delta t_i a_i)^{n_i} x_i(k) \\
&+ \sum_{j=1}^{n_i} (1+\Delta t_i a_i)^{j-1} \Delta t_i b_i u_i(k)
\end{aligned} \tag{3.25}$$

The sum in (3.25) can be recognized as a known geometric progression,

$$\sum_{j=1}^{n} q^{j-1} = \frac{1-q^n}{1-q}, \tag{3.26}$$

and if $a_i \neq 0$

$$\sum_{j=1}^{n_i} (1+\Delta t_i a_i)^{j-1} = \frac{(1+\Delta t_i a_i)^{n_i} - 1}{\Delta t_i a_i} \tag{3.27}$$

By inserting $k := t$, where t is the progressing time, and $k + n_i = t + T_d$, the global solution of the solved subsystem can be expressed as

$$\begin{aligned}
x_i(t+T_d) &= (1+\Delta t_i a_i)^{n_i} x_i(t) \\
&+ \left[(1+\Delta t_i a_i)^{n_i} - 1\right] \frac{b_i}{a_i} u_i(t)
\end{aligned} \tag{3.28}$$

These results are similar to the solution of discrete-time equations given in [29], and for comparison the solution for a continuous-time state equation with sampled input is given as

$$\begin{aligned}
x_i(t+T_d) &= e^{a_i T_d} x_i(t) \\
&+ \left(\int_0^{T_d} e^{a_i (T_d - \tau)} d\tau\right) b_i u_i(t) \\
&= e^{a_i T_d} x_i(t) + \left(e^{a_i T_d} - 1\right) \frac{b_i}{a_i} u_i(t)
\end{aligned} \tag{3.29}$$

It can be seen that when $\Delta t_i \to 0$, (3.28)→≈(3.29) where n_i is given as in (3.23), when neglecting higher order terms in the Tailor expansion of the exponential function. By including the output mapping given as in (3.2), we might for simplicity rewrite subsystem i in (3.28) as

$$\begin{aligned} x_i(t + T_d) &= a_{n_i} x_i(t) + b_{n_i} u_i(t) \\ y_i(t) &= c_i x_i(t) \end{aligned} \tag{3.30}$$

where

$$\begin{aligned} a_{n_i} &:= \begin{cases} (1 + \Delta t_i a_i)^{n_i} & \text{for } a_i \neq 0 \\ 1 & \text{for } a_i = 0 \end{cases} \\ b_{n_i} &:= \begin{cases} \frac{b_i}{a_i}(a_{n_i} - 1) & \text{for } a_i \neq 0 \\ T_d b_i & \text{for } a_i = 0 \end{cases} \end{aligned} \tag{3.31}$$

Equivalently, when a subsystem contains a set of differential equations, we may rewrite (3.30) as

$$\begin{aligned} \boldsymbol{x}_i(t + T_d) &= \boldsymbol{A}_{n_i} \boldsymbol{x}_i(t) + \boldsymbol{B}_{n_i} \boldsymbol{u}_i(t) \\ \boldsymbol{y}_i(t) &= \boldsymbol{C}_i \boldsymbol{x}_i(t) \end{aligned} \tag{3.32}$$

where

$$\begin{aligned} \boldsymbol{A}_{n_i} &:= (\boldsymbol{I} + \Delta t_i \boldsymbol{A}_i)^{n_i} \\ \boldsymbol{B}_{n_i} &:= \boldsymbol{A}_i^{-1}(\boldsymbol{A}_{n_i} - \boldsymbol{I})\boldsymbol{B}_i, \quad \boldsymbol{A}_i \text{ is nonsingular.} \end{aligned} \tag{3.33}$$

and where $\boldsymbol{x}_i \in \mathcal{R}^m$, $\boldsymbol{A}_i \in \mathcal{R}^{m \times m}$, $\boldsymbol{u}_i \in \mathcal{R}^p$, $\boldsymbol{B}_i \in \mathcal{R}^{m \times p}$, $\boldsymbol{y}_i \in \mathcal{R}^r$ and $\boldsymbol{C}_i \in \mathcal{R}^{r \times m}$, meaning that there are m states, p inputs and r outputs in subsystem i. If $\boldsymbol{A}_{n_i}$ is singular, similar requirement as established in (3.31) can be applied. Note that if the subsystem output is also dependent on the input vector, $\boldsymbol{y}_i = \boldsymbol{C}_i \boldsymbol{x}_i + \boldsymbol{D}_i \boldsymbol{u}_i$, then $\boldsymbol{D}_{n_i} = \boldsymbol{D}_i$, and is included when mapping all subsystems into a total system.

Now, let us assume that a given distributed system contains s linear subsystems, with solutions given as either (3.30) or (3.32). These subsystems are connected together in the total distributed co-simulation system through a predefined connection configuration, typically

$$\boldsymbol{u}_d = \boldsymbol{M}_d \boldsymbol{y}_d \tag{3.34}$$

where $\boldsymbol{u}_d$ is a vector containing inputs for all subsystems, $\boldsymbol{y}_d$ is a vector containing outputs for all subsystems and $\boldsymbol{M}_d$ is a mapping matrix. Then, we might define the global solution of the distributed system as

$$\boldsymbol{x}_d(t + T_d) = (\boldsymbol{A}_d + \boldsymbol{B}_d)\boldsymbol{x}_d \tag{3.35}$$

where $\boldsymbol{x}_d$ denotes all states in the distributed system,

$$\boldsymbol{A}_d := \text{diag}(\boldsymbol{A}_{n_1}, \boldsymbol{A}_{n_2}, ..., \boldsymbol{A}_{n_s}) \tag{3.36}$$

and $\boldsymbol{B}_d$ is a mapping matrix between outputs and inputs with the diagonal equal to zero. Since $\boldsymbol{A}_d$ and $\boldsymbol{B}_d$ have the same size we might define the solution mapping matrix as

$$\boldsymbol{S}_d = \boldsymbol{A}_d + \boldsymbol{B}_d \tag{3.37}$$

such that

$$\boldsymbol{x}_d(t + T_d) = \boldsymbol{S}_d \boldsymbol{x}_d \tag{3.38}$$

To clarify, if the solution mapping matrix for a distributed system containing two single linear differential equations such as in (3.2) where each differential equation is implemented as a separate subsystem, the solution mapping matrix is given as

$$\boldsymbol{S}_d = \begin{bmatrix} a_{n_1} & b_{n_1} c_2 \\ b_{n_2} c_1 & a_{n_2} \end{bmatrix} \tag{3.39}$$

where a_{n_i} and b_{n_1} is defined as in 3.31. Note that a similar expression for $\boldsymbol{S}_d$ can be found when $y_i(t) = c_i x_i(t) + d_i u_i(t)$, where d_i is assumed a constant mapping value in system i.

In order for a general distributed system to have a stable global solution, we must first assure that

$$|\boldsymbol{S}_d| \leqslant 1 \tag{3.40}$$

by choosing T_d. In other words, we must choose T_d such that all eigenvalues in $\boldsymbol{S}_d$ have an amplitude with absolute value less than, or equal to, 1. However it is actually not enough to guarantee that the solution mapping matrix has decreasing characteristics. This has to do with possible differences between the local subsystem propagating times and the global propagating time and will be discussed more in detail later on. The derived stability criterion is tested in the following Example.

Example 3.3 (Stabilization of linear distributed system)**.** *Consider two single linear differential subsystems given as*

$$\begin{aligned} \dot{x}_1 &= -2x_1 + u_1 \\ y_1 &= x_1 \end{aligned} \tag{3.41}$$

and

$$\begin{aligned} \dot{x}_2 &= -1.5x_2 + 0.5u_2 \\ y_2 &= x_2 \end{aligned} \tag{3.42}$$

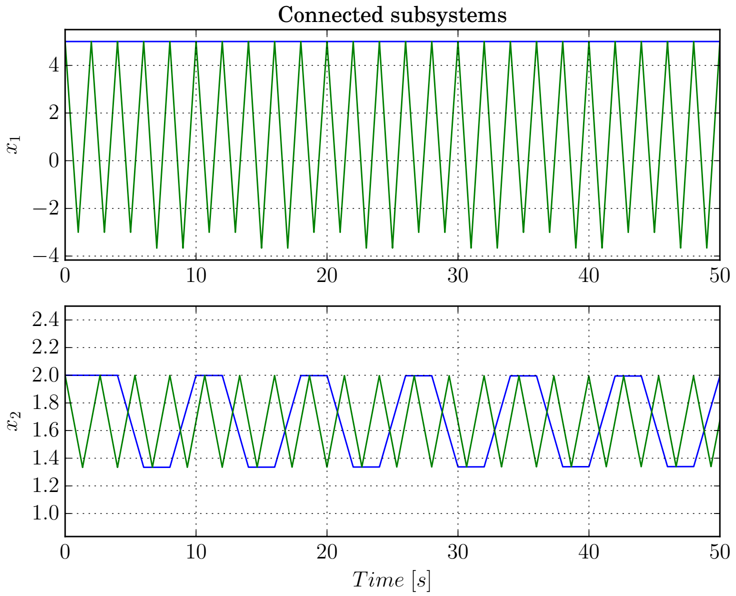


Figure 3.7: Simulation results of two subsystems, with marginally stable local solvers, connected in a distributed system. Green graph denotes local solution and blue graph denotes global solution. $\Delta t_1 = 1.0\,\mathrm{s}$, $\Delta t_2 = 1.333\,\mathrm{s}$, $T_d = 2.0\,\mathrm{s}$

which are connected in a distributed simulation such that $u_1 = y_2 = x_2$ and $u_2 = y_1 = x_1$. Assume that each of the subsystems are solved by the Euler integration method. By setting the initial conditions $x_1(0) = 5$ and $x_2(0) = 2$ we can write the solution mapping matrix for the system as

$$\boldsymbol{S}_d = \begin{bmatrix} (1 - 2\Delta t_1)^{n_1} & -\frac{1}{2}[(1 - 2\Delta t_1)^{n_1} - 1] \\ -\frac{1}{3}[(1 - 1.5\Delta t_2)^{n_2} - 1] & (1 - 1.5\Delta t_2)^{n_2} \end{bmatrix} \tag{3.43}$$

It can be verified that by setting $\Delta t_1 = 1.0\,\mathrm{s}$ and $\Delta t_2 = \frac{4}{3}\,\mathrm{s}$, meaning that the local solvers are only marginally stable, the absolute values of the two eigenvalues in $\boldsymbol{S}_d$ are equal to 1 when $T_d = 2.0\,\mathrm{s}$. This gives $n_1 = 2$ and $n_2 = 1.5$ which means that subsystem 1 is allowed 2 local steps and subsystem 2 is allowed 1.5 local steps between each data exchange. This means that subsystem 2 does not always have its local propagating time synchronized with the global propagating time, which may introduce errors causing the distributed system to become unstable. Figure 3.7 shows the simulation results for the two connected subsystems.

The first plot in Figure 3.7 shows the global solution, blue graph, compared to the local solution, green graph, for subsystem 1. Due to the sampling frequency, the

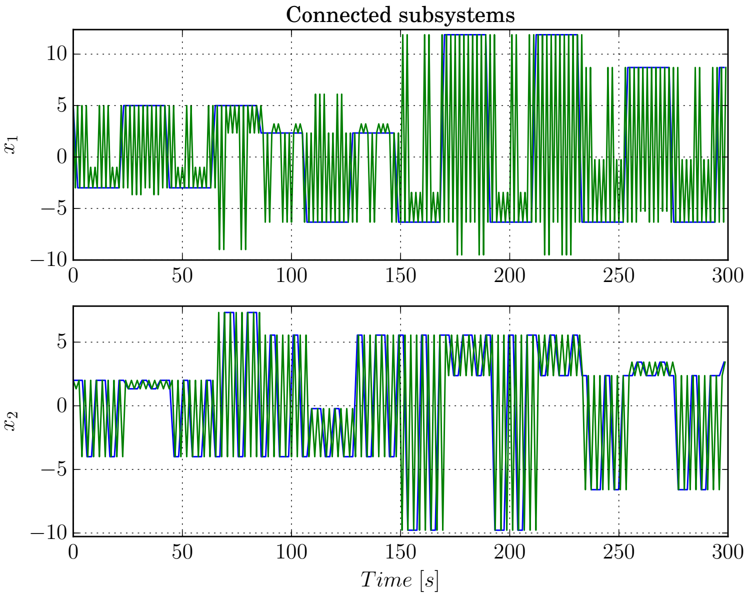


Figure 3.8: Simulation results of two subsystems, with marginally stable local solvers, connected in a distributed system. Green graph denotes local solution and blue graph denotes global solution. $\Delta t_1 = 1.0\,\mathrm{s}$, $\Delta t_2 = 1.333\,\mathrm{s}$, $T_d = 2.1\,\mathrm{s}$

communication frequency between the subsystem, the global solution is constant in comparison to the local solution which oscillates and is barely affected by subsystem 2. This is as expected when the global time-step size is chosen twice as large as the local time-step size and when the local solver is marginally stable. However the local propagating time is always synchronized with the global propagating time. The second plot in Figure 3.7 shows a rather different situation. Here, the global solution oscillates and with a different frequency than the local solution. However, the total system is stable because the global solution of subsystem 1 is constant and does not excite subsystem 2 in any particular way other than giving an offset.

To illustrate more in detail the effect of unsynchronized propagating times the global time step is changed to $T_d = 2.1\,\mathrm{s}$ s which gives $|eig(\boldsymbol{S}_d)| = \{0.9877,\ 0.9560\}$. Figure 3.8 shows the corresponding simulation results. Clearly, the total distributed system is unstable, which indicates that synchronized propagating time is closely related to the simulation stability in this setting. ◇

Example 3.3 shows that synchronization of the propagating times is crucial for global solution stability. It can also be shown in the derivation of the solution mapping matrix where it is actually assumed that the times are synchronized without giving it any thoughts. Based on the derived stability criterion and the experiences gained in Example 3.3 the following theorem can be established.

Theorem 3.1 (Convergence of local and global solutions in distributed systems.)**.** *Given a set of linear subsystems in a distributed system, each given as*

$$\begin{aligned} \dot{\boldsymbol{x}}_i &= \boldsymbol{A}_i \boldsymbol{x}_i + \boldsymbol{B}_i \boldsymbol{u}_i \\ \boldsymbol{y}_i &= \boldsymbol{C}_i \boldsymbol{x}_i \end{aligned} \tag{3.44}$$

that are solved locally by fixed step size solvers, such as the forward Euler integration method with a time step Δt_i, and by assuming that each local propagating time is synchronized with the global propagating time, the global solution of subsystem i can be expressed as

$$\begin{aligned} \boldsymbol{x}_i(t + T_d) &= \boldsymbol{A}_{n_i} \boldsymbol{x}_i(t) + \boldsymbol{B}_{n_i} \boldsymbol{u}_i(t) \\ \boldsymbol{y}_i(t) &= \boldsymbol{C}_i \boldsymbol{x}_i(t) \end{aligned} \tag{3.45}$$

where $\boldsymbol{A}_{n_i}$ and $\boldsymbol{B}_{n_i}$ are solver dependent matrices (for the forward Euler integration method see (3.33)*) where*

$$n_i := \frac{T_d}{\Delta t_i} \tag{3.46}$$

and T_d is the global step size in the distributed system. By collecting the subsystems and applying the given connection setup, the total global solution of the distributed system may be expressed as

$$\boldsymbol{x}_d(t + T_d) = \boldsymbol{S}_d(\boldsymbol{A}_{n_i}, \boldsymbol{B}_{n_i}, \boldsymbol{C}_i)\boldsymbol{x}_d(t) \tag{3.47}$$

where $\boldsymbol{S}_d(\cdot)$ is denoted the solution mapping matrix for the total distributed system. Then, if T_d can be chosen such that

$$|eig(\boldsymbol{S}_d(\cdot))| \leqslant 1 \tag{3.48}$$

then both the local and the global solution will be stable. Moreover, if

$$|eig(\boldsymbol{S}_d(\cdot))| < 1 \tag{3.49}$$

the global steady state solution of the distributed system will converge to the local steady state solutions.

Proof. The proof is equal to the derivation of $\boldsymbol{S}_d(\cdot)$, where the forward Euler integration method is used, when assuming synchronized propagating times. The proof is also similar when other explicit integration methods are used. □

Definition 3.1. *A distributed system is said to have synchronized local and global propagating times either when $n_i \in \mathcal{N}_{\geqslant 1}$ for each subsystem i, where $\mathcal{N}_{\geqslant 1}$ denotes all integers larger than, or equal to 1, or if local time-step size control that assure time synchronization is implemented in each subsystem in a co-simulation.*

As can be seen in Theorem 3.1 we do not need to specify that each subsystem is locally stable, both in the sense of dynamical stability and numerical stability. This has to do with the fact that both the solver characteristics and the system dynamics are already included in the stability criterion. Example 3.4 shows an application of Theorem 3.1.

Example 3.4 (Example 3.3 continued)**.** *Assuming a distributed system is given as in Example 3.3, but now $\Delta t_1 = 0.9$, $\Delta t_2 = 0.9$ and $T_d = 1.8$, giving $n_1 = n_2 = 2$. Note that these local time steps guarantees that the local solutions will be stable by themselves when using the Euler integration method. Then the amplitudes of the eigenvalues in the solution mapping matrix are given as $|eig(\boldsymbol{S}_d)| = \{0.7271,\ 0.0354\}$. Since $|eig(\boldsymbol{S}_d)| < 1$ and $n_i \in \mathcal{N}_{\geqslant 1}\ \forall i$, the steady state global solution will converge to the local solutions according to Theorem 3.1. The simulation results are shown in Figure 3.9. As can be seen in the figure the steady state global solution converge to the two steady state local solutions. Hence, the total system solution is stable.* ◇

As can be seen in Example 3.4 the global solution converges to the local solutions when the propagating local times are synchronized with the global propagating time and when the magnitude of the eigenvalues of $\boldsymbol{S}_d < 1$. However, Theorem 3.1 may be relaxed with respect to time synchronization when the local solvers are robust, even though it is not recommended due to accuracy reasons as illustrated in Example 3.3.

Until now only distributed co-simulation systems containing subsystems with stable dynamics and stable local solvers have been studied. However Theorem 3.1 provides a method for analysing both numerical and dynamical stability in a distributed system without certain passivity properties as long as the propagating times are synchronized. It is also possible to determine stability limits for a distributed system containing one or more subsystems that have unstable dynamics and dependent on surrounding stabilizing systems in order for the total distributed simulation to become stable. This is shown in Example 3.5.

Example 3.5 (Unstable Subsystem)**.** *Assume that two linear single differential systems connected in a distribute simulation are given as*

$$\begin{aligned} \dot{x}_1 &= -x_1 + u_1 \\ \dot{x}_2 &= 0.1x_2 - u_2 \end{aligned} \tag{3.50}$$

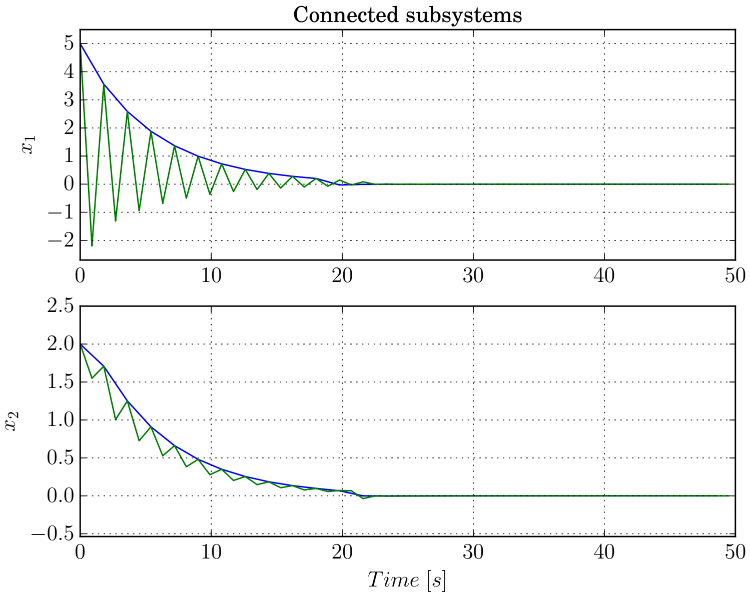


Figure 3.9: Simulation result of two subsystems with stable local solvers connected as in a distributed system. Green graph denotes local solution and blue graph denotes global solution.

and where $y_1 = x_1$ and $y_2 = x_2$. As before, at each communication point $u_1 = y_2$ and $u_2 = y_1$. Subsystem 2 is clearly unstable, but it can be verified that the eigenvalues for the continuously connected systems are given as $\lambda_{1,2} = \frac{-9 \pm 3i\sqrt{31}}{20}$, and is therefore dynamically stable. However, when solving the two systems separately in a distributed manner, subsystem 2 would become unstable when T_d becomes arbitrarily large. The Euler integration method is used to solve both systems locally, and the time steps are set to $\Delta t_1 = \Delta t_2 = 0.01$. It can be verified that when choosing $T_d = 0.89$, giving $n_1 = n_2 = 89$, $|eig(\boldsymbol{S}_d)| = \{0.9984,\ 0.9984\}$. Hence, according to Theorem 3.1 the total distributed system simulation results are stable. The simulation results are shown in Figure 3.10.

◇

3.4.3 Explicit Fixed Step Size Solvers

So far, only the forward Euler integration method has been studied. However, in [34] also higher order explicit numerical solvers were studied, and similar stability conditions were found. These are summarized in Table 3.1. The following example illustrates the use of two different solvers applied on a mass-damper-spring

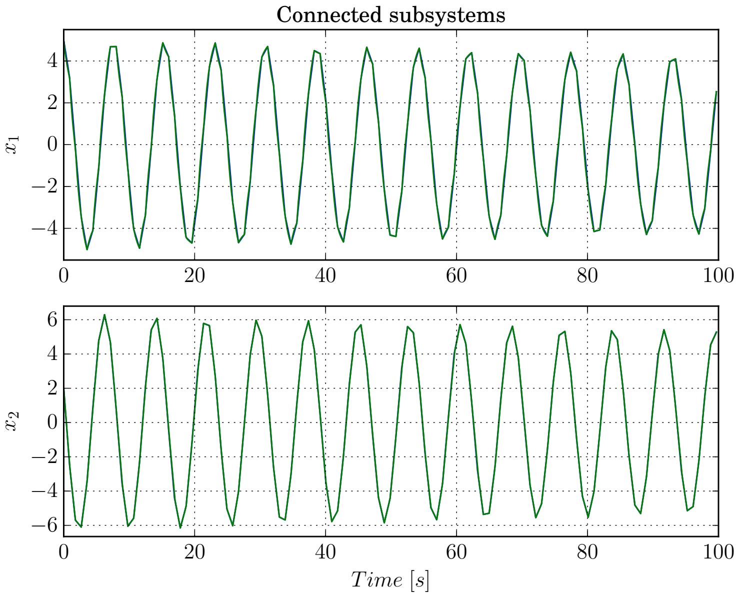


Figure 3.10: Simulation result of connected subsystems in a distributed system where one of the subsystems has unstable dynamics. Green graph denotes local solution and blue graph denotes global solution.

Table 3.1: Summary of solver dependent matrices.

Solver	$\boldsymbol{A}_{n_i}$	$\boldsymbol{B}_{n_i}$	
forward Euler	$\tilde{Exp}(\Delta t_i \boldsymbol{A}_i, 1)^{n_i}$	$\boldsymbol{A}_i^{-1}(\boldsymbol{A}_{n_i} - \boldsymbol{I})\boldsymbol{B}_i,$	$\boldsymbol{A}_i$ not singular
RK2	$\tilde{Exp}(\Delta t_i \boldsymbol{A}_i, 2)^{n_i}$	$\boldsymbol{A}_i^{-1}(\boldsymbol{A}_{n_i} - \boldsymbol{I})\boldsymbol{B}_i,$	$\boldsymbol{A}_i$ not singular
RK3	$\tilde{Exp}(\Delta t_i \boldsymbol{A}_i, 3)^{n_i}$	$\boldsymbol{A}_i^{-1}(\boldsymbol{A}_{n_i} - \boldsymbol{I})\boldsymbol{B}_i,$	$\boldsymbol{A}_i$ not singular
RK4	$\tilde{Exp}(\Delta t_i \boldsymbol{A}_i, 4)^{n_i}$	$\boldsymbol{A}_i^{-1}(\boldsymbol{A}_{n_i} - \boldsymbol{I})\boldsymbol{B}_i,$	$\boldsymbol{A}_i$ not singular

system controlled by a speed regulator.

Example 3.6 (Distributed mass-damper-spring system with speed regulator)**.** *A mass-damper-spring system as shown in Figure 3.11 is to be implemented as a subsystem in a distributed system, and the differential equations describing its dynamics are given as*

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}(F_r - bx_2 - kx_1) \end{aligned} \tag{3.51}$$

where x_1 is the position, x_2 is the speed and F_r is the speed regulator force. This

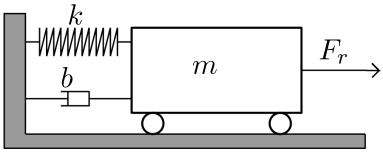


Figure 3.11: Mass-damper-spring system.

regulator force is assumed to be another subsystem in the distributed system given as

$$\dot{F}_r = K_I F_r + K_P x_2 \tag{3.52}$$

and is comparable to a PI-controller where the speed reference is set to zero. We might assume that the mass-damper-spring dynamics are solved with the Euler integration method and the regulator with the RK2 integration method. Hence, we might write

$$\boldsymbol{S}_d = \begin{bmatrix} \boldsymbol{A}_{n_1} & \boldsymbol{B}_{n_1} \\ 0, b_{n_2} & a_{n_2} \end{bmatrix} \tag{3.53}$$

where

$$\boldsymbol{A}_{n_1} = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \Delta t_1 \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \right)^{n_1} \tag{3.54}$$

$$\boldsymbol{B}_{n_1} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix}^{-1} \left(\boldsymbol{A}_{n_1} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \tag{3.55}$$

$$a_{n_2} = \left(1 + K_I \Delta t_2 + \frac{K_I^2}{2} \Delta t_2^2 \right)^{n_2} \tag{3.56}$$

and

$$b_{n_2} = (a_{n_2} - 1) \frac{K_P}{K_I} \tag{3.57}$$

where n_i is as defined in (3.23)*. By setting $m = 2$, $k = 10$, $b = 1$, $K_I = -0.1$, $K_P = -0.5$, $\Delta t_1 = 0.01$ and $\Delta t_2 = 0.1$, it can be verified that $T_d = 1.0$ gives $n_1 = 100$, $n_2 = 10$ and $|eig(\boldsymbol{S_d})| = \{0.8424,\ 0.8424,\ 0.9075\}$, which means that the distributed system is stable when solved. The initial values are set as $x_1(0) = 5$, $x_2(0) = 0$ and $F_r(0) = 2$, and the simulation results are shown in Figure 3.12.*

It should be mentioned that the control law would have preformed much better if $K_P > 0$ which would have put F_r and x_2 in opposite phases, not equal as shown in the figure. Nevertheless, a negative value for K_P was used in order to make the system oscillate more an make the system less robust. ◇

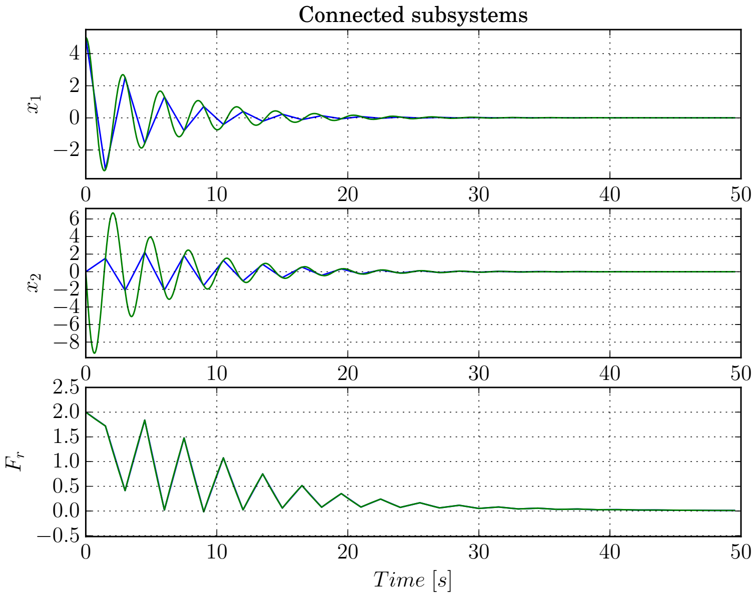


Figure 3.12: Simulation results showing the speed controlled mass-damper-spring system. Green graph denotes local solution and blue graph denotes global solution.

As seen in the example, it can be more time consuming to calculate the solution matrix when using higher order explicit solvers in comparison to first order solvers such as the forward Euler integration method, at least when the total system becomes large. However, if only a conservative stability result is required it is possible to assure stability by using a lower order solver in the analysis as long as the stability regions of the solvers overlap. This is summarized in Corollary 3.1.

Corollary 3.1. *A conservative stability analysis can be performed for a distributed system containing higher order solvers by reducing the order of the solvers. For example, a conservative stability result can be found for distributed systems containing higher order explicit Runge-Kutta integration methods by assuming that the forward Euler integration method is used in the simulation.*

Proof. Since the higher order explicit Runge-Kutta integration methods contain the stability region for the forward Euler integration method, see Figure 3.1, simulation results from a distributed system containing higher order Runge-Kutta integration methods would be stable if stability is assured when using the forward Euler integration method. □

3.4.4 Towards Non-linear Systems

So far, much attention has been given to distributed linear dynamical systems. However, as it turns out, the numerical stability criterion for linear dynamical systems can also be applied to non-linear systems, although generating more conservative results. To illustrate this, assume that a non-linear dynamical system is given as

$$\begin{aligned} \dot{x} &= -x^3 + u \\ y &= x + u \end{aligned} \tag{3.58}$$

which is globally exponentially stable when $u = 0$. It can be shown that the linearized system is given as

$$\begin{aligned} \dot{\Delta x} &= -3x_0^2 \Delta x + u \\ x &= x_0 + \Delta x \\ y &= x + u \end{aligned} \tag{3.59}$$

where x_0 is the operating point for the linearized system. The linearized system has a range of eigenvalues, given as a function of x_0,

$$\lambda(x_0) = -3x_0^2 \tag{3.60}$$

which in this case only contains negative eigenvalues. This is of no surprise since the non-linear dynamics are stable when the input is set to zero. It is then possible to specify a validity range of the non-linear system, such that the system is only valid for a finite range of values for x, typically $x \in [x_l, x_u]$ where the subscript $_l$ stands for the lower limit and subscript $_u$ stands for the upper limit. If $x_l = -x_u$ for the system given in (3.58), the eigenvalues for the system are in the range $\lambda \in [-3x_u^2, 0]$. This means that when using for example the Euler integration method, a simulation of the non-linear system will be stable as long as $\Delta t \leqslant \frac{2}{3x_u^2}$.

In general, for a non-linear system given as

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ \boldsymbol{y} &= \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}) \end{aligned} \tag{3.61}$$

the linearized system can for the operation points $\boldsymbol{x}_0$, $\boldsymbol{u}_0$ and $\boldsymbol{y}_0$ be expressed as

$$\begin{aligned}
\dot{\Delta\boldsymbol{x}} &= \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0\\ \boldsymbol{u}=\boldsymbol{u}_0}} \Delta\boldsymbol{x} + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0\\ \boldsymbol{u}=\boldsymbol{u}_0}} \Delta\boldsymbol{u}\\
\Delta\boldsymbol{y} &= \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0\\ \boldsymbol{u}=\boldsymbol{u}_0}} \Delta\boldsymbol{x} + \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_0\\ \boldsymbol{u}=\boldsymbol{u}_0}} \Delta\boldsymbol{u}\\
\boldsymbol{x} &= \boldsymbol{x}_0 + \Delta\boldsymbol{x}\\
\boldsymbol{u} &= \boldsymbol{u}_0 + \Delta\boldsymbol{u}\\
\boldsymbol{y} &= \boldsymbol{y}_0 + \Delta\boldsymbol{y}
\end{aligned} \tag{3.62}$$

Note that both the inputs, the outputs and the states need to be linearized in order to apply the stability criterion. Also note that the control vector $\boldsymbol{\tau}_c$ is omitted in (3.62) since the control law can be treated as internal system dynamics or a separate distributed subsystem. Corollary 3.2 summarizes the procedure for applying Theorem 3.1 for analysing stability in non-linear distributed systems.

Corollary 3.2 (Simulation Stability Criterion for Non-linear Systems)**.** *A non-linear dynamical system as given in* (3.61) *can be linearized according to* (3.62) *and if finite ranges for the states, the inputs and the outputs can be determined and given as $\boldsymbol{x} \in [\boldsymbol{x}_l, \boldsymbol{x}_u]$, $\boldsymbol{u} \in [\boldsymbol{u}_l, \boldsymbol{u}_u]$ and $\boldsymbol{y} \in [\boldsymbol{y}_l, \boldsymbol{y}_u]$, respectively, Theorem 3.1 can be applied where $\boldsymbol{A}_i(\boldsymbol{x}_{i,0})$, $\boldsymbol{B}_i(\boldsymbol{u}_{i,0})$ and $\boldsymbol{C}_i(\boldsymbol{y}_{i,0})$ are used to determine conservative values for Δt_i and T_d that make the total distributed simulation stable.*

Proof. A linearized version of a non-linear system i, given as in (3.62), can be expressed as

$$\begin{aligned}
\dot{\Delta\boldsymbol{x}}_i &= \boldsymbol{A}_i(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})\Delta\boldsymbol{x}_i + \boldsymbol{B}_i(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})\Delta\boldsymbol{u}_i\\
\Delta\boldsymbol{y}_i &= \boldsymbol{C}_i(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})\Delta\boldsymbol{x}_i + \boldsymbol{D}_i(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})\Delta\boldsymbol{u}_i
\end{aligned} \tag{3.63}$$

Given $\boldsymbol{x}_{0i} \in [\boldsymbol{x}_{li}, \boldsymbol{x}_{ui}]$, $\boldsymbol{u}_{0i} \in [\boldsymbol{u}_{li}, \boldsymbol{u}_{ui}]$ and $\boldsymbol{y}_{0i} \in [\boldsymbol{y}_{li}, \boldsymbol{y}_{ui}]$ then $\exists$ eigenvalues for system i such that $\boldsymbol{\lambda}_i(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i}) \in [\boldsymbol{\lambda}_{li}, \boldsymbol{\lambda}_{ui}]$, which means that a finite range of eigenvalues can be determined. Moreover, $\exists$ $\boldsymbol{A}_{n_i}(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})$, $\boldsymbol{B}_{n_i}(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})$, $\boldsymbol{C}_{n_i}(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})$ and $\boldsymbol{D}_{n_i}(\boldsymbol{x}_{0i}, \boldsymbol{u}_{0i})$ such that

$$\text{eig}(\boldsymbol{S}_d(\boldsymbol{A}_{n_1}, \boldsymbol{B}_{n_1}, \cdots, \boldsymbol{D}_{n_N})) \in [\text{eig}(\boldsymbol{S}_d)_l, \text{eig}(\boldsymbol{S}_d)_u] \tag{3.64}$$

Then, if both $|\text{eig}(S_d)_l| \leqslant 1$ and $|\text{eig}(S_d)_u| \leqslant 1$, the distributed simulation results including the non-linear system are stable. □

Typically the finite ranges for the states, the inputs and the outputs are chosen based on validity regions of the subsystems, initial values, maximal expected values or saturation limits. To illustrate the use of Corollary 3.2, Example 3.6 is revised.

Example 3.7 (Regulated and distributed mass-damper-spring system with non–linear spring)**.** *The mass-damper-spring system given in Figure 3.11 has now a spring with non-linear spring stiffness given as kx^2. Hence, the system of differential equations is expressed as*

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}(F_r - bx_2 - kx_1^3) \end{aligned} \tag{3.65}$$

The linearized mass-damper-spring system can be written as

$$\begin{aligned} \Delta\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}(F_r - bx_2 - 3kx_{01}^2\Delta x) \end{aligned} \tag{3.66}$$

which gives

$$\boldsymbol{A}_{n_1}(x_{01}) = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \Delta t_1 \begin{bmatrix} 0 & 1 \\ -\frac{3kx_{01}^2}{m} & -\frac{b}{m} \end{bmatrix}\right)^{n_1} \tag{3.67}$$

$$\boldsymbol{B}_{n_1}(x_{01}) = \begin{bmatrix} 0 & 1 \\ -\frac{3kx_{01}^2}{m} & -\frac{b}{m} \end{bmatrix}^{-1} \left(\boldsymbol{A}_{n_1} - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right) \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} \tag{3.68}$$

Since the uncontrolled system is strictly passive and since the regulator was able to stabilize the linear mass-damper-spring system in Example 3.6, it is reasonable to assume that the initial value for x_1 in (3.66) *would be the highest value for x_1 such that $x_1 \in [-x_1(0), x_1(0)]$. By assuming that all values are as in Example 3.6, it can be verified that $|eig(\boldsymbol{S}_d)| = \{1.0000, 1.0000, 0.9922\}$ when $\Delta t_1 =$ 0.0013 s, $\Delta t_2 =$ 0.026 s and $T_d =$ 0.078 s, giving $n_1 = 60$ and $n_2 = 3$. The simulation results are shown in Figure 3.13. As can be seen in the figure the total distributed system is stable and converges relatively quickly to zero. This means that the stability criterion for non-linear systems, as given in Corollary 3.2 is a bit conservative. However, this is not surprising since maximal values for the states, the inputs and the outputs were used in the stability criterion.* ◇

3.5 Accuracy Control

When representing physical systems as mathematical models in a co-simulation we introduce discrete system interaction events because of the discrete global communication points. In a real physical system such subsystem interactions are continuous which means that the discrete interactions in a co-simulation may introduce

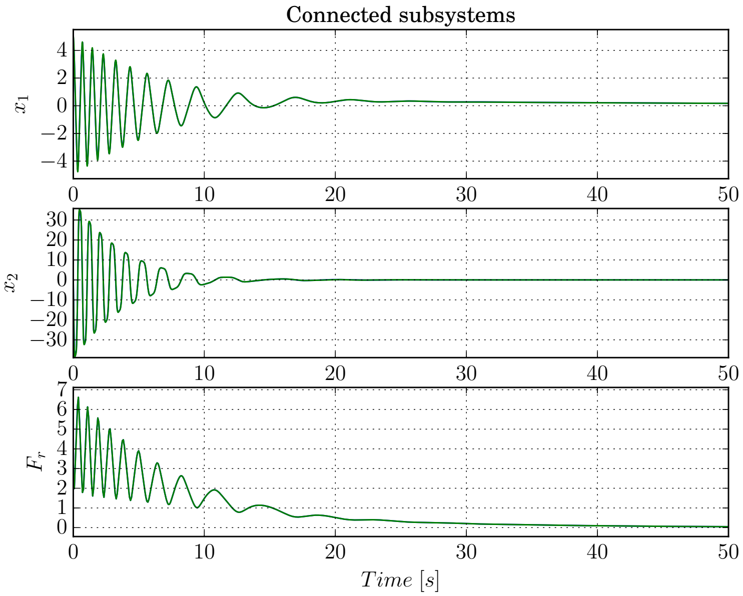


Figure 3.13: Simulation results from regulated and distributed mass-damper-spring system with non-linear spring stiffness. Green graph denotes local solution and blue graph denotes global solution

room for significant inaccuracies, all depending on the dynamics of the system, e.g. the ability to dissipate energy, the choice of T_d and how the subsystem inputs are treated between the communication points. For example, we saw in Figure 2.3 that FOH method seems to have a better representation of the inputs than the ZOH method, which would result in a better accuracy than when keeping the inputs constant between communication points. However, when decreasing T_d significantly, the total co-simulation system approaches a continuous system representation.

The reason for often choosing a high global communication time step in co-simulations is split. Often a fast solvable system is preferred, and by increasing T_d it is possible to also increase the local solver time steps as long as the total co-simulation system remains stable. Also, T_d should be chosen such that $T_d \geqslant \max(\Delta t_i)\ \forall i \in N$, where N is the number of local solvers in a co-simulation, in order to enable time synchronization for all subsystems. Since a co-simulation can contain subsystems with quite different dynamics, giving the co-simulation a range of dynamical time constants, the global communication time step will often be chosen high for some of the subsystems, leading to inaccuracies in the co-simulation results.

When discussing accuracy in simulations in general, one is faced to address the question of what are accurate simulation results. This question might have many different answers depending on the angle of reflection. Hence, when talking about continuous systems we might divide accuracy into *model accuracy* and *numerical accuracy*, where the former is a measure of how good a model can reproduce the characteristics of a physical system, and the latter a measure of numerical convergence in numerical solvers, loosely speaking. However, when considering distributed simulations, this division is not sufficient because of the discrete communication points. A subsystem in a co-simulation might be accurate in representing the dynamics of a physical system, but because of the co-simulation characteristics it becomes inaccurate because a large communication time step has been chosen. Hence, we should also include the term *co-simulation accuracy*, which reflects the effects on the simulation results when having a large global communication time step.

Here, we are only interested in the co-simulation accuracy, since model accuracy is something the model developer should answer for, and since numerical accuracy is related to numerical stability and convergence, e.g. how good the simulation results fit the exact solution of the system if such is available. The co-simulation accuracy is a measure of wrongly setting the inputs to subsystems in a co-simulation. For example, when considering the ZOH method, the co-simulation accuracy is a measure of the effect of keeping the inputs constant between the communication points. When using power variables as inputs and outputs between subsystems in a co-simulation, we can also say that co-simulation accuracy is the measure of supplying a subsystem with the wrong amount of energy between communication points. Hence, according to Figure 2.3 the FOH method has a higher co-simulation accuracy than the ZOH method, generally speaking.

In order for a co-simulation to be efficient at the same time as reaching a pre-set threshold of co-simulation accuracy, one must control the size of T_d in an efficient manner. As discussed in [35] there exists many methods for doing this, but only a few that do not require simulator-internal data or the ability to revert to a previous communication time step and redo the entire global communication time step, often referred to as *rollback*. However, one method for controlling the global communication time-step size is proposed in [110, 80], which adaptively controls the global communication time-step size by the use of a non-iterative predictor-corrector error estimator. Another method for controlling the co-simulation accuracy is presented in [35] which adaptively controls the communication time-step size by comparing the difference in power exchanged through the subsystem connections, hence, utilizing the basic principles of bond graph theory. A thorough presentation of this method is given in [35]. Thus, a more practical presentation of the method is given here.

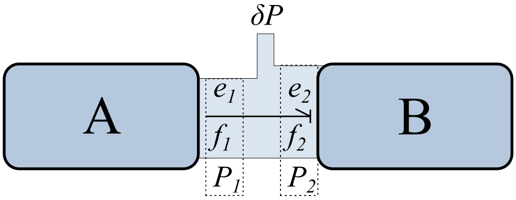


Figure 3.14: Power flow between two subsystems in a co-simulation

Consider two connected subsystems in a co-simulation as shown in Figure 3.14. The power can be calculated for each subsystem as

$$P_i = e_i f_i \tag{3.69}$$

Hence, the difference in power in the connection between the two subsystems in the figure can be calculated as

$$\delta P_k = P_2 - P_1 \tag{3.70}$$

where k represent the connection number. Note that a slightly different definition is given in [35], but here we have assumed a positive power flow direction, from system **A** to **B**. Correspondingly, the wrongfully added energy between two communication points can be expressed as

$$\delta E_k(t + Td) = \int_t^{t+T_d} \delta P_k(t)dt \approx \delta P_k(t + T_d)T_d \tag{3.71}$$

The average energy exchanged between the two subsystems can be calculated as

$$E_k(t) = \frac{1}{2}\left(P_1(t) + P_2(t)\right)T_d \tag{3.72}$$

When having K number of closed loop connections in a co-simulation, a scalar error indicator can according to [35] be defined as

$$\epsilon(t) = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\left(\frac{\delta E_k(t)}{r_k(E_{0k} + |E_k(t)|)}\right)^2} \tag{3.73}$$

where E_{0k} is a typical energy scale and r_k is a relative tolerance, both for connection k. A PI-controller is proposed to adaptively adjust the global communication time step and is given as

$$T_{d,i+1} = \alpha_s \epsilon(t_i)^{-k_I - k_P} \epsilon(t_{i-1})^{k_P} T_{d,i} \tag{3.74}$$

where k_I and k_P are the integral gain, $0 < \alpha_s < 1$ is a safety factor and i is the communication point number. In addition to this control law, user-defined limits for T_d can be defined, $T_{d,min} \leqslant T_d \leqslant T_{d,max}$, as well as rate limits of T_d, $\Theta_{min} \leqslant T_{d,i}/T_{d,i-1} \leqslant \Theta_{max}$.

To illustrate the use of adaptive step-size control in co-simulations, an example is given in the following.

Example 3.8 (Accuracy control through adaptive communication time-step size). *Consider again the co-simulation system given in* (3.18)*. Even though we have not defined the inputs and outputs as power variables, the method can be applied directly although we are not longer talking about energy. The parameters in the adaptive step size controller is set to* $E_{01} = 100$, $r_1 = 0.000001$, $\alpha_2 = 0.8$, $T_{d,0} = 0.1\,\mathrm{s}$, $T_{d,min} = \Delta t_1 = \Delta t_2 = 0.001\,\mathrm{s}$, $T_{d,max} = 1.5\,\mathrm{s}$, $k_I = 0.1$, $k_P = 0.2$, $\Theta_{min} = 0.2$ *and* $\Theta_{max} = 1.5$. *Note that the maximal value of* T_d *is set higher than the stable maximal value derived in section 3.3.3 in order to test the method also outside the stability region. As before, the initial conditions are set to* $x_1(0) = 5$ *and* $x_2(0) = 2$*. The simulation results, along with the exact solution of the continuous version of the system, are shown in Figure 3.14.*

The two first upper lefthand-most plots show the results regarding the two states in the co-simulation, both local and global solution, and are compared to the exact solution of the system which is obtained when treating the total system as a continuous system. Corresponding magnified regions for the two plots are shown to the right. The last plot shows the value of T_d *during the entire co-simulation. As seen in the figure both the local and the global co-simulation results for the two states seem to converge to the exact solution. This has to do with the low pre-set initial value for* T_d*. After the transient phase in the co-simulation, when the states seem to be stabilized at zero,* T_d *is increased by the adaptive step-size control law. Even though* T_d *overshoots the stable value at first, which was calculated in Example 3.1, it converges to the same value after some time. Note that the magnified regions of the states show that the states oscillate a bit around zero. This, because the pre-set relative tolerance enables the step-size controller to increase the communication time step such that it becomes marginally stable (*$T_d = 1.111\,\mathrm{s}$*) before all the energy is dissipated in the entire system. However, these oscillations are small and we can conclude that the adaptive step-size controller works properly in increasing the accuracy of the simulation results while maximizing the global communication time-step size in the co-simulation. It may seem like the relative tolerance is not reached since both amplitudes of the state oscillations are larger than the tolerance. However, this relative tolerance represent energy, not the state itself. Here, it means that the tolerance must be less than, or equal to, the product of the two states, which it is.* ◇

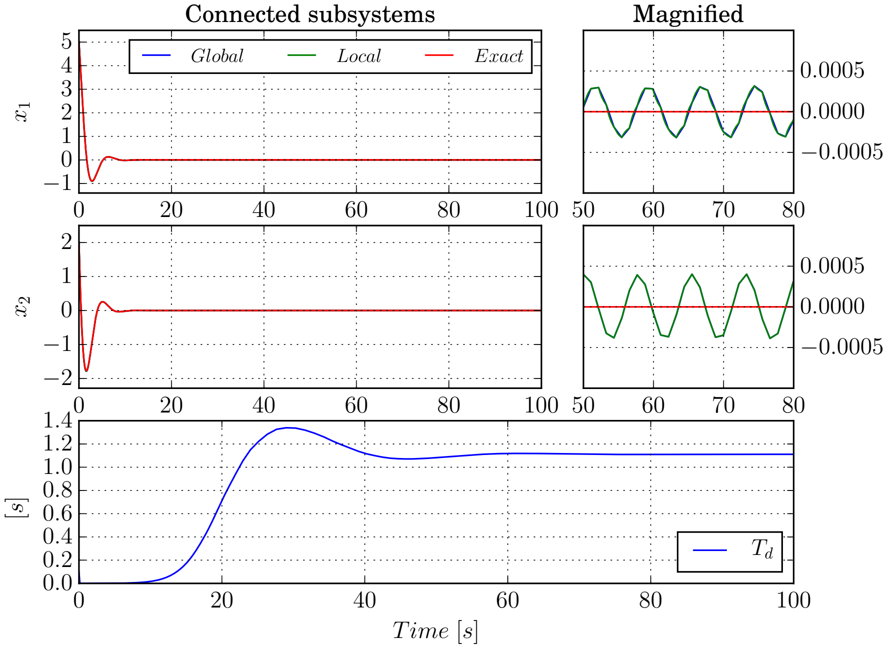


Figure 3.15: Co-simulation results when using adaptive global communication time-step size control

When using adaptive global communication time-step size control one must include functionalities that assure synchronization of all propagating times in the co-simulation. Hence, in comparison to the co-simulation algorithm presented in Algorithm 2.1 a size check of local solver time-step sizes are included, and if $t_i + \Delta t_i \geqslant k_i T_d$, where t_i is the propagating time in subsystem i, Δt_i is the corresponding local time step and k_i is a communication point counter, then the corresponding local time-step size are set to $\Delta t_i = k_i T_d - t_i$.

This adaptive communication time-step size controller is in [35] compared to another explicit method for controlling the communication time-step size [110, 80] using a quarter-car model as a benchmark model.

3.6 Conclusion

In this chapter the stability and accuracy of distributed simulations have been discussed. A short literature review of applicable dynamical stability theories has been conducted and the effect the global communication time step has on the eigenvalues in a co-simulation system has been discussed. The main focus concerning stability has been given to numerical stability and linear distributed co-simulation systems where the numerical stability has been studied by exploring outer solver stability limits. Numerical stability of distributed simulations have not been treated in any detail in the literature. In this chapter stability requirements for explicit solvers for differential equations have been combined with system dynamics and the global communication time-step in distributed system, and a criterion for guaranteeing stable simulations has been derived. This criterion is also extended to include non-linear system dynamics in the stability analysis. Several examples are given to illustrate the use of the criterion as well as illustrating stable numerical results from distributed co-simulations. In practice, both dynamical stability analyses and numerical stability analyses are quite time consuming when the co-simulation system grows, both with the increase in number of subsystems as well as in states. Also, when excluding passive systems, all subsystem informations must be known on beforehand which excludes black-box models from such analyses. Therefore, in the rest of this work, stability analysis of co-simulation systems are excluded and stability of systems are evaluated in practice through co-simulation results.

Closely related to numerical stability in a co-simulation is the accuracy of the simulation results themselves. The concept of accuracy has been discussed in relation to co-simulations and an explicit adaptive communication time-step size controller has been presented for controlling the co-simulation accuracy. This, in order to obtain more accurate simulation results as well as keeping the time it takes to solve the co-simulation system at a minimum.

CHAPTER 4

Tightly Coupled Systems and Hybrid Causality Models

This chapter is based on a restructured and reformatted version of [37, see **P5** in section 1.6] where a generic and mathematical approach for reformulating differential algebraic equations into differential equations based on the idea of adding small, possible negligible, dynamical contributions, are presented and analysed. Such differential algebraic equations often may also come as a result of changing causality in order to assure connectivity in a system. This mathematical approach enables hybrid causality configurations for models which increases the possibility of connecting a model to a given model environment without introducing algebraic loops through the connections themselves. Also, the hybrid causality model framework enables standard dynamical stability analysis theories as well as simulator stability theory for distributed systems [34, 36, see **P2** and **P4** in section 1.6, respectively], as discussed in chapter 3. It also enables the use of explicit numerical solvers since differential algebraic equations are absent, although this would also depend on other internal system dynamics.

4.1 Introduction

When making mathematical model representations of real physical systems or processes it may in some cases be difficult to choose the level of model fidelity. It is well known that it is nearly impossible to represent a real physical system or process perfectly with a mathematical model, but one can get pretty close. However, often a "good enough" modeling approach is aimed for when studying only the most significant dynamics in a system. The choice of model fidelity may set restrictions later on when connecting a model to a larger model environment. These restrictions are usually related to connectivity, overall system stability, solver stability and available computational power. One might define some standard inputs and outputs for generic domain models, assuring connectivity on a general level, but connectivity can not be guaranteed for all possible options.

On the other hand, some systems are hard to split for co-simulation purposes because they are closely linked to each other, mathematically speaking, and are often referred to as *tightly coupled*. This type of systems include systems with algebraic dependencies and systems that excite each other with special frequencies. Tightly coupled systems are in general not suited for distribution but can be distributed if treated correctly. Here, we divide tightly coupled systems into two groups, namely systems that are tightly coupled through *frequencies* and systems that are tightly coupled through *causality*, where the latter will be in the main focus in this chapter.

4.1.1 Tightly Coupled through Frequencies

A system that is sensitive to input signals, which often include fast dynamics, is said to be tightly coupled to the system environment through frequencies, in contrast to passive systems as discussed in section 3.3.2. Hence, additional care must be taken when distributing such systems. One example of such a tightly coupled system is a three phase generator powering an electrical motor, both modelled as subsystems in a distributed system by using the (a, b, c)-reference frame [39]. Then, by assuming that the model interfaces are voltages and currents, these interface signals oscillate with the operating frequency of the power system, typically 50-60 Hz in steady state, in addition to other transient frequencies. According to the Nyquist sampling theorem, the two subsystems should exchange data with a frequency of atleast 100-120 Hz in order to avoid aliasing. However, in practice the frequency of the data exchange should be much higher in a distributed system in order to obtain stable and realistic simulation results. Thus, such a system is tightly coupled through frequencies and eigenvalues in a co-simulation if the rest of the subsystems in the co-simulation can communicate with a much lower frequency. Such tightly coupled systems can be distributed if both local numerical solvers, the corresponding solver time-step sizes and the global communication time-step size are chosen properly[1].

Typically, electrical systems modelled in the (a, b, c)-reference frame are considered to be tightly coupled through frequencies. However, if another reference frame is used, such as the $(d, q, 0)$-reference frame [111], the electrical system is not considered tightly coupled, unless other dynamical effects makes the system sensitive, because the signals connecting the subsystems do not oscillate with the power grid frequency. Such systems are discussed more in detail in Chapter 6.

[1] In the sense of stability and sampling theorems.

4.1.2 Tightly Coupled through Causality

Some systems are said to be tightly coupled through causality, the theory of *action* and *reaction* in- and between dynamics [26], which means that they are closely connected through the differential equations, where only one of the systems can have a full state-space implementation, whereas the other systems must include differential algebraic equation(s), reducing the number of states in each system. This, in order to assure connectivity between the systems and to avoid iterations on a higher level in a distributed co-simulation system. One typical example of a system that is tightly coupled through causality in the field of maritime industry is a marine vessel with a heavy deck crane as will be discussed in chapter 5. The vessel and the crane have in general large time constants, which means that they are not tightly coupled through frequencies, but they are still tightly coupled, thus through causality.

Systems that are tightly coupled through causality are not distributable right away because they can not be connected directly, as will be elaborated further throughout this chapter. This type of tightly coupled system offers several serious problems regarding co-simulations where the common factor seems to be differential algebraic equations, as will be discussed in the following.

4.2 Background and Motivation

Before taking a deep dive into the core topics presented in this chapter, practical examples of causality configurations, background theory, motivation and a review of relevant literature are presented in the following.

4.2.1 Causality Configurations

A first order differential equation with only one single input and one single output (SISO) has two causality orientations, namely *integral causality* and *differential causality*. Integral causality is the mathematical representation of a system that produces differential equations, while the differential causality representation of s system also produces one or more differential algebraic equation(s). The two different causality options are illustrated in Example 4.1.

Example 4.1 (Causality Configurations)**.** *Consider a linear 1 degree of freedom (DOF) pendulum consisting of a mass, a damper and a spring, only affected by an external force $F(t)$, as shown in Figure 4.1. The spring stiffness is given as k, the damping coefficient as b, the mass as m and the external force as $F(t)$. The system dynamics can be expressed as a well-known second order differential equation given as*

$$m\ddot{x} + b\dot{x} + kx = F(t) \tag{4.1}$$

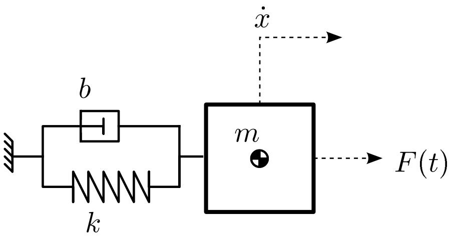


Figure 4.1: Mass-damper-spring system

and a set of first order differential equations given as

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{m}(F(t) - bx_2 - kx_1) \end{aligned} \tag{4.2}$$

where x_1 and x_2 are the position and the velocity, respectively.

The same system can be formulated with differential causality and expressed as

$$m\frac{dv(t)}{dt} + bv(t) + k\int_0^t v(t)dt = F(x_1, t) \tag{4.3}$$

or simply

$$\begin{aligned} \dot{x}_1 &= v(t) \\ y = F(x_1, t) &= m\frac{dv(t)}{dt} + bv(t) + kx_1 \end{aligned} \tag{4.4}$$

where $F(x_1, t)$ is the forces given as a reactive feedback to the forced velocity. As can be seen, the state vector has been changed in comparison to the previous causality form and the number of states are reduced from two to one. ◇

The example presented in this section gives a short introduction to model causality forms. As can be seen, only integral causality gives the complete set of first order differential equations and is often the preferred choice of causality if one is able to choose. However in some cases the modeller is not able to choose integral causality without introducing the need of iteration because of inexplicit connectivity. Hence, there exists both advantages and disadvantages related to choosing one causality over the other, as will be discussed in the following.

4.2.2 Related Work and Motivation

The connectivity of a system is closely related to the structure of the mathematical implementation, the causality of the model [25], as discussed in section 4.2.1. By choosing the causality of a model, the mathematical structure and implementation of the model is explicitly given, as long as the level of model fidelity is held constant. A model with differential causality, which contains at least one differential algebraic equation, must be treated more carefully than a model with complete integral causality. This, because numerical errors when solving the system become of greater concern, if not algebraic solutions are available, and lots of analysis theories for analysing dynamical stability, as the ones discussed in section 3.3, are disregarded because of the differential algebraic equations.

Causality and Connectivity

When modeling a complete system as one total stand-alone model, connectivity is not that important since the system information is globally available in the total model. This means that in many cases differential causalities can be solved numerically, or even analytically which always is the case for linear systems [112, 113]. Moreover, if an algebraic solution of the differential causality problem is not available, many numerical solvers have the ability to perform iterations such that the differential causalities can be solved numerically [83]. This is however more troublesome in explicit distributed co-simulation schemes.

In general, a subsystem in a co-simulation can be solved numerically in a stable manner if the correct numerical solver is used [114, 115, 116], no matter which causalities it contains, as long as the original non-distributed model contains stable dynamics. However, often it is preferred to avoid re-stepping and iteration between subsystems in a distributed system, e.g. due to possible real-time criteria if for example hardware is to be interfaced with the simulation as in [41]. Also, re-stepping and iterations between subsystems in a distributed system require well defined and sophisticated functionalities in the co-simulation master algorithm, which few master algorithms have, in addition to possible exchange more information between the subsystems than only the subsystem outputs, e.g. such as Jacobi matrices and signal rates. Thus, in practice this means that between global time steps the submodels should be independent of each other. This implies that algebraic solutions of differential algebraic equations, as well as numerical solutions of differential algebraic equations, dependent on the input-output model mapping (I/O-mapping), should not be considered. Hence, one could be tempted to force integral causality on all subsystems, but then problems regarding connectivity occur, as illustrated in Example 4.2.

Example 4.2 (Causality-Connectivity Loop)**.** *Assume that two of the mass-damper-spring system presented in Example 4.1 are to be connected to each other in a distributed system by assuming that the masses have the same motion, as illustrated in figure 4.2. If both systems are set to have integral causality, the system setup*

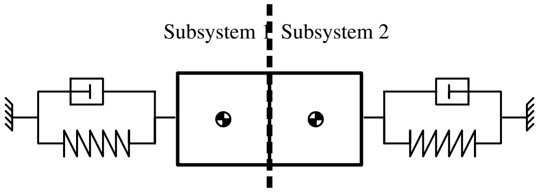


Figure 4.2: Connected Mass-damper-spring systems with loop caused by connectivity problems

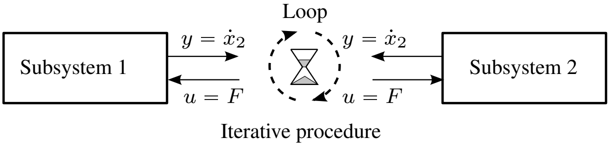


Figure 4.3: Connectivity issue between two mass-damper-spring systems with integral causality

would be as shown in Figure 4.3. As can be seen in the figure, both submodels give velocities as outputs and receive forces as inputs, meaning that the connectivity of the submodels fails. This cause a loop outside the submodel environments and iterations are needed in order to solve the problem. ◇

As shown in the example, the causality-connectivity problem causes an iterative loop between the two submodels. One could be tempted to change the causality of one of the systems and calculate the derivative of the input signal numerically, but this is a bad idea for distributed systems where the signals between the models appear as sampled signals. Also, since differential causality is used for one of the models one still cannot apply the standard stability analysis theories for assuring model- and total system stability.

Example 4.2 may seem like a system only constructed for illustrating the causality-connectivity problem. However, it is closely related to the interpretation of many real-life systems, one being a crane attached to a marine vehicle, where both the crane and the vehicle are implemented as subsystems with complete integral causality forms. Even though such systems often have many degrees of freedom, there are methods for combining these subsystems into one system with only integral causalities. This is elaborated in [38] and in chapter 5 for a crane attached to a marine vessel. However, this method contradicts model modularity, and it is not straight forward to replace the crane design with another.

Dynamic Causality

The causality-connectivity problem escalates and gets even more interesting when allowing submodels to change dynamics online during a simulation, due to discrete time event approximations such as e.g. closing and opening an electrical circuit, or in order to extend the connectivity options. This feature is relevant for both distributed systems and non-distributed systems and is often referred to as model switching. The reason for allowing such a feature in a model could also be for allowing failure dynamics, e.g. provide simulation results of a realistic system containing a failing component. This may force a switching of causality in the model, changing the input-output (I/O) mapping of the model. In general, this often leads to loosing a differential equation, since it is often the case that an integral causality, a differential equation, gets changed to differential causality, a differential algebraic equation. Models with the ability to switch between causalities online during a simulation and, hence, change the I/O mapping are here referred to as *hybrid causality models* and is one of the main topics presented in this chapter.

Much work on switched systems in general is documented in the literature, from many different angles, ranging from pure mathematics and electrical system theory to applied modeling and control theory. In [90, 91] a mathematical framework describing hybrid system dynamics in the sense of continuous and discrete time events is presented and has its advantages when it comes to system stability analysis and model-based control designs. This framework has some similarities with Tavernini's model [117] and Back-Guckenheimer-Myers model [118]. Other examples of hybrid model frameworks are the Nerode-Kohn model [119] and Brockett's model [120], to mention a few. A thorough review of these hybrid models are given in [121]. The main similarity between the mentioned hybrid frameworks is that they all depend on a state-space formulation that only consists of differential equations.

In the field of bond graph modeling theory, there also exists a few contributions to hybrid system's theory. In [122] a modeling method for mode switching in systems using bond graph theory is presented and in [123] switched power junctions are presented and enable hybrid dynamics in bond graph theory. Even though mode switching and switched power junctions using bond graph theory enable the use of differential algebraic equations, the hybrid systems must be tailored in each case and do not enable commonly used dynamical stability analysis tools.

When modeling a non-distributed system in a commercial modeling and simulation software the causalities are often solved automatically by the software before a simulation is performed. However, if simulation events that change the system dynamics are considered, most commercial modeling and simulation soft-

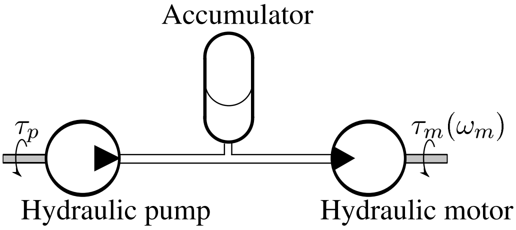


Figure 4.4: Hydraulic power system with accumulator

ware fail to construct the state-space model, hence, not being able to solve the system without special workarounds, such as reinitialization of the numerical solver, and in some cases also changing the integration method. Hence, this would require that each distributed subsystem with the ability to switch causality need to be implemented with a tailored solver structure and mapping of initial conditions dependent on the previous causality configuration, which would in general become quite complex even for simple linear systems. This is illustrated in Example 4.3.

Example 4.3 (Hydraulic Power System)**.** *Assume that a hydraulic power system with an almost incompressible hydraulic fluid, containing a hydraulic pump, an accumulator and a loaded hydraulic motor is as shown in Figure 4.4, with τ_p constant, and a linear motor loading, τ_m, only dependent on the motor speed ω_m. Furthermore, assume that there are no leakages in the system. The fluid flows between the pump and the accumulator and the motor and the accumulator are given as $\dot{Q}_1$ and $\dot{Q}_2$, respectively, and the pressure in the system is given as p. It can be shown that this system can be modeled with complete integral causality, having $\dot{Q}_1$, $\dot{Q}_2$ and p as states.*

It is further assumed that if the accumulator runs dry it gets damaged and fails. In this situation, the total system starts acting as a hydraulic power system without the accumulator. Then, one of the other hydraulic components must provide the pressure in the pipeline while the remaining system a fluid flow. This will result in differential causality for one of the models and thereby a reduction in states, in addition to the one lost due to the absent accumulator. In general, the system has now one state in comparison to the hydraulic power system with a working accumulator that has three states. ◇

As the example shows, special simulation events might trigger changes in the state space model of a system, and if a change in causality is triggered the total state-space form is affected due to a change in number of internal states. In order to overcome such problems in commercial modeling and simulation software, often model switching and integrator restarting, or passive system states in the state space model, are considered in order to solve the system. Hence, additional system logics are needed for controlling the total state space model in both cases and need be tailored in each case.

Towards a Solution

In the field of applied mathematical modeling, such differential algebraic equations in a model are often changed to differential equations by adding a small dynamic contribution, such as compliance effects. This, if an algebraic solution to the differential causality problem can not be found with ease as in [112, 124] for non-distributed systems. However, adding small dynamic effects does not come without a cost. Such small effects often tend to stiffen the system, requiring suited solvers for stiff systems [115, 116], or at least a reduction in solver time-step size, if not treated carefully. An advantage with this approach is that the number of states in the system is kept constant despite changing the causality, since one can activate or deactivate the added dynamics in a generic fashion, and if a subsystem contains a set of differential equations, only the differential algebraic equations need to be changed.

Potential internal differential causalities in a model are related to the causality-connectivity problem between models since by moving the local model boundaries such that the total system becomes one model, the causality-connectivity problem becomes an internal differential causality problem. Hence, both problems can be solved by the same solution, if such a solution exists. The common factor in Example 4.2 and 4.3 is the differential causality and if the differential algebraic equations, in general, could be changed to differential equations in a generic and robust way without affecting the connectivity, most of the already stated problems would vanish, although new ones might occur. This also seems to be a simpler task than implementing special and generic functionalities in a distributed simulation master algorithm, or to tailor a model switching simulation scenario. However, one must be careful not to affect the internal dynamics too much such that the *reformulated differential causality model* gives the same results as the model containing differential algebraic equation(s). This is studied in more detail in the following based on adding a low-pass filters with derivative effects to the differential algebraic equations such that they become differential equations.

4.3 Differential Causality

As one might understand from a mathematical point of view, integral causality is the preferred causality form in model representations of real systems since it produces differential equations. However the differential causality form can be the preferred choice in distributed systems when the connectivity between subsystems fails, as discussed in section 4.2.2. The other choice would be to solve the loop caused by connectivity problems outside the subsystem environment as shown in Figure 4.3, which can be quite time consuming and requires additional functionalities in the distributed simulation master.

Example 4.3 illustrated that different dynamical effects in a model have predefined inputs and outputs from the integral causality. This might be fine if the causality orientation is static, but when simulation events that change the causality online during a simulation are considered, problems occur, as discussed in section 4.2.2. Hence, since the differential causality is the common factor both in causality-connectivity problems in distributed simulations and causality changes made by simulation events in non-distributed systems a new formulation of the differential causality can be the common solution. As mentioned in section 4.1, a commonly used solution for differential algebraic equations can be found in the field of engineering, and is discussed in the following.

4.3.1 Added Dynamics

When the connectivity between two integral causality models fails, such as in the mechanical system shown in Figure 4.2, the engineering approach for ensuring connectivity is to add small and preferably negligible dynamical effects that retain the integral causality while ensuring connectivity. These dynamical effects often consist of capacitive and dissipative effects, such as a spring and a damper in mechanical systems or an RC-circuit in electrical systems, and hence, contributes to additional states in the total state-space models, instead of reducing the number of states as would be the case if differential causality was considered. This is illustrated in Example 4.4.

Example 4.4 (Added Dynamical Effects)**.** *Consider the mechanical system given in Example 4.2 but now a spring and a damper is placed between the two masses. By assuming that the outputs from the two mass-damper-spring systems are the velocities of the masses, the difference in velocity can be defined as*

$$\Delta v := v_1 - v_2 \tag{4.5}$$

Assume that the additional spring-damper system between the two masses has a spring stiffness k_a and a damping coefficient b_a. Then it can be shown that the force generated by the spring-damper system can be expressed as

$$F_a = k_a \int_0^t \Delta v dt + b_a \Delta v \tag{4.6}$$

The added spring-damper system works as a Proportional-Integral (PI) controller for the difference between the velocities and will try to make them converge ($\Delta v =$

0*) by feeding the two subsystems with a control force* F_a*. The transfer function for this control law can be expressed as*

$$\frac{\Delta v}{F_a}(s) = \frac{s\frac{1}{k_a}}{\frac{b_a}{k_a}s + 1} \equiv \frac{k_f s}{T_f s + 1} \tag{4.7}$$

where $k_f = \frac{1}{k_a}$ *is the transfer function gain and* $T_f = k_f b_a$ *is the transfer function time constant. Note that if an electrical system is considered,* $k_a = \frac{1}{C}$ *and* $b_a = R$ *which give the well known time constant* $T_f = RC$*. Also note that if the positions of the masses were considered as model outputs, the control law would change to a Proportional-Derivative (PD) control law, which explains why there always becomes a small difference between the two positions of the masses when applying this method.* ◇

As shown in Example 4.4, the iteration loop outside the two submodels in Figure 4.2 can be avoided by designing a control law that makes the model outputs converge, although this adds another state to the total system. When considering changing causality online during a simulation, it would be preferred to retain the number of states in the system constant due to implementation reasons since no passive states are needed.

Based on the PI-control law added through a simple spring-damper system in Example 4.4, a generic method for reformulating differential causality into differential equations is presented in the following.

4.3.2 Reformulation of Port Dependent Differential Causalities

Since a differential causality removes one state from the total state-space model in a system, it is believed that negligible dynamical effects can be added both in order to regain the lost state and to assure connectivity.

Consider a single linear differential equation equation, and its output, given as

$$\begin{aligned} \dot{x} &= ax + bu \\ y &= hx \end{aligned} \tag{4.8}$$

where x is the state, u is the input, y is the output and a, b and h are constants. The differential causality form of (4.8) is given as

$$\begin{aligned} x &= \frac{y}{h} \\ u &= \frac{1}{b}\left(\dot{x} - ax\right) \end{aligned} \tag{4.9}$$

Now, y is given as input and u as output in the system, and the system has been reduced with one state. For simplicity in syntax, lets define the input to the differential causality form $u^* := y$ and the corresponding output as $y^* := u$, which gives

$$\begin{aligned} x &= \frac{u^*}{h} \\ y^* &= \frac{1}{b}\left(\dot{x} - ax\right) \end{aligned} \tag{4.10}$$

This system does not contain any differential equations, hence, common stability theories can not be applied. Also, simple explicit numerical solvers can not be applied due to the $\dot{x}$ term present in the system equations. However, the solution for regaining the lost differential equation and the state may actually be related to this term. If we could find a method for differentiating x that introduces a differential equation, the differential causality is transformed into integral causality without changing the inputs and outputs of a model.

By looking more closely at the transfer function in (4.7), it can also be interpreted as a low-pass filter with derivative effects with a filter gain equal to k_f. Moreover, it can be seen that for frequencies lower than the cut-off frequency $f_c = \frac{1}{2\pi T_f}$, the transfer function can be approximated as

$$\frac{\Delta v}{F_a}(s) \approx k_f s \tag{4.11}$$

and hence,

$$\Delta v \approx k_f \frac{dF_a}{dt} \tag{4.12}$$

when solved in the time domain. Hence, it is possible to regain the state that is lost due to the differential causality by applying a low-pass filter with derivative effect as long as the cut-off frequency is set properly. The low-pass filtering effects also help smoothing the sampled inputs which enables numerical differentiation of the signals without tremendous peaks. However, as (4.12) shows, we must set $k_f = 1$. Therefore, from a dynamical point of view, the low-pass filter time constant T_f is only affected by the chosen damping coefficient. Hence, the transfer function we seek is given as

$$H(s) = \frac{s}{Ts + 1} \tag{4.13}$$

where T is the time constant for the band limit and $f_c = \frac{1}{2\pi T}$ is the band limit frequency, or the low-pass filter cut-off frequency. By defining $H(s) = \frac{z(s)}{\sigma(s)}$, where $\sigma(s) := \frac{u^*(s)}{h}$ and where $z(s)$ is the approximated value for $\dot{x}$, we might rewrite the transfer function as

$$\begin{aligned}(Ts+1)z(s) &= s\sigma(s) \\ \Rightarrow z(s) &= \frac{1}{T}\sigma(s) - \frac{1}{Ts}z(s)\end{aligned} \tag{4.14}$$

which can be transformed to a state space model as

$$\begin{aligned}\dot{\xi} &= \frac{1}{T}\left(\frac{u^*}{h} - \xi\right) \\ z &= \frac{1}{T}\left(\frac{u^*}{h} - \xi\right)\end{aligned} \tag{4.15}$$

where $z := \dot{x}$ when neglecting the initial conditions. This means that we have regained one state, ξ, and correspondingly, one differential equation. By inserting (4.15) in (4.10) the new differential causality system can be rewritten as

$$\begin{aligned}\dot{\xi} &= -\frac{1}{T}\xi + \frac{1}{Th}u^* \\ y^* &= -\frac{1}{Tb}\xi + \frac{1}{bh}(\frac{1}{T} - a)u^*\end{aligned} \tag{4.16}$$

The transfer function based differentiation introduce a few new topics that need to be studied, such as initial conditions when switching between the causality models, solver stability and convergence. These topics are addressed in the following.

4.3.3 Initial Conditions and Solver Stability

When deriving the state space model for the transfer function given in (4.13) the initial conditions were neglected, because these can be set after each discrete time event where causality switching is initiated. In the systems studied here, the power should be constant in all discrete events, *jumps*, e.g. no power is produced or dissipated. This means that if a jump from the integral causality to the differential causality model happens at time $t = t_0$, $y^*(t_0) = u(t_0)$. This is achieved by setting the initial condition

$$\xi(t_0) = (1 - aT)x(t_0) - Tbu(t_0) \tag{4.17}$$

and if a jump from the differential causality model to the integral causality model happens at time $t = t_0$, $y(t_0) = u^*(t_0)$, the initial condition is given as

$$x(t_0) = \frac{u^*(t_0)}{h} \tag{4.18}$$

If the model contains more than one differential equation, the initial conditions can be found similarly.

When it comes to solver stability, the stability will be affected by the parameters T and a. Solver stability for distributed simulations is thoroughly treated in chapter 3 and will only be briefly mentioned here. By assuming that the two causality models given in (4.8) and (4.16) are to be implemented and solved with the Euler integration method, the local solver time-step size must be set according to

$$\Delta t \leqslant 2 \cdot \min \left(T, \frac{1}{|a|} \right), \quad a < 0 \tag{4.19}$$

in order to provide a stable numerical solution. This means that the time constant in the transfer function must be considered in addition to a when choosing Δt, assuming that the same solver and solver time-step size is used for both causality configurations. In general, one could be tempted to choose

$$T > \frac{1}{|\mathrm{Re}(a)|} \tag{4.20}$$

for models with only one differential equation when assuming that $a < 0$. However, this would be a poor choice since the dynamics of the transfer function then would become slower than the originally dynamics in the physical model. This means that one must be careful when choosing T, and it is believed that the choice of T would both affect the choice of Δt and the convergence of the transfer function based differentiation to $\dot{x}$. This is studied in more detail in the following.

4.3.4 Convergence of Transfer Function based Differentiation

To study the convergence of the transfer function based differentiation z given in (4.15) to $\dot{x}$ it would be smart to remove the effect of the solver error. This can be done by solving the differential equations algebraically. The system given in (4.8) has an algebraic solution given as

$$x(t) = e^{at}x(t_0) + \int_0^t e^{a(t-\tau)} bu(\tau) d\tau \tag{4.21}$$

when assuming that $t_0 = 0$. By setting u constant, the solution is given as

$$x(t) = e^{at}x(0) + (e^{at} - 1)\frac{b}{a}u \tag{4.22}$$

and the output is given as before, $y(t) = hx(t)$. The same can be done for (4.16), and when assuming that the input is given as $u^*(t) = y(t) = hx(t)$, the solution can be written as

$$\begin{aligned} \xi(t) = {} & e^{-\frac{t}{T}}\xi(0) + \left(\frac{\xi(0)}{aT+1} + \frac{bu}{a^2T+a} \right) \left[e^{at} - e^{-\frac{t}{T}} \right] \\ & + \frac{b}{a}u \left(e^{-\frac{t}{T}} - 1 \right) \end{aligned} \tag{4.23}$$

and the output $y^*(t)$ is given as

$$\begin{aligned} y^*(t) =& -\frac{1}{Tb}e^{-\frac{t}{T}}\xi(0) \\ & -\frac{1}{Tb}\left[\left(\frac{\xi(0)}{aT+1}+\frac{bu}{a^2T+a}\right)\left[e^{at}-e^{-\frac{t}{T}}\right]\right] \\ & +\frac{1}{b}\left(\frac{1}{T}-a\right)\left[e^{at}x(0)+\left(e^{at}-1\right)\frac{b}{a}u\right] \\ & -\frac{1}{Ta}\left(e^{-\frac{t}{T}}-1\right)u \end{aligned} \tag{4.24}$$

It can be seen that

$$\lim_{t\to\infty} y^*(t) = (1-\frac{1}{Ta})u+\frac{1}{Ta}u = u \tag{4.25}$$

which means that the the transfer function based differentiation of x converges to $\dot{x}$ when the solution is not affected by numerical errors, u is constant and the rate of convergence is dependent on T and a. This is also almost true when $u = u(t)$ as long as the band limit time constant is chosen such that the band limit frequency is higher than the largest frequency in $u(t)$ and faster than the fastest dynamics in the model. Such convergence results are shown in Example 4.5-4.7.

Example 4.5 (Algebraic Solution)**.** *A mathematical model representation of a physical system is given as*

$$\begin{aligned} \dot{x} &= -5x+u \\ y &= x \end{aligned} \tag{4.26}$$

where $u = 1.0,\ \forall t \in [0, 2.0]$ *and* $x(0) = 0$*. The solution of this mathematical model representation is given as*

$$x(t) = e^{-5t}x(0)+\frac{1}{5}(1-e^{-5t}) \tag{4.27}$$

As can be seen, $\lim_{t\to\infty} x(t) = 0.2$*. The solution* $y(t) = x(t)$ *is given as input to a reformulated differential causality version of the system,* $u^*(t) = y(t)$*, given as*

$$\begin{aligned} \dot{\xi} &= -\frac{1}{T}\xi+\frac{1}{Th}u^* \\ y^* &= -\frac{1}{Tb}\xi+\frac{1}{bh}\left(\frac{1}{T}-a\right)u^* \end{aligned} \tag{4.28}$$

where $T = 0.001$ *s,* $h = 1.0$*,* $b = 1.0$*,* $a = -5.0$ *and the solution of* $y^*(t)$ *is given as in* (4.24)*. The initial condition of* ξ *is set to* $\xi(0) = -0.001$*, according to*

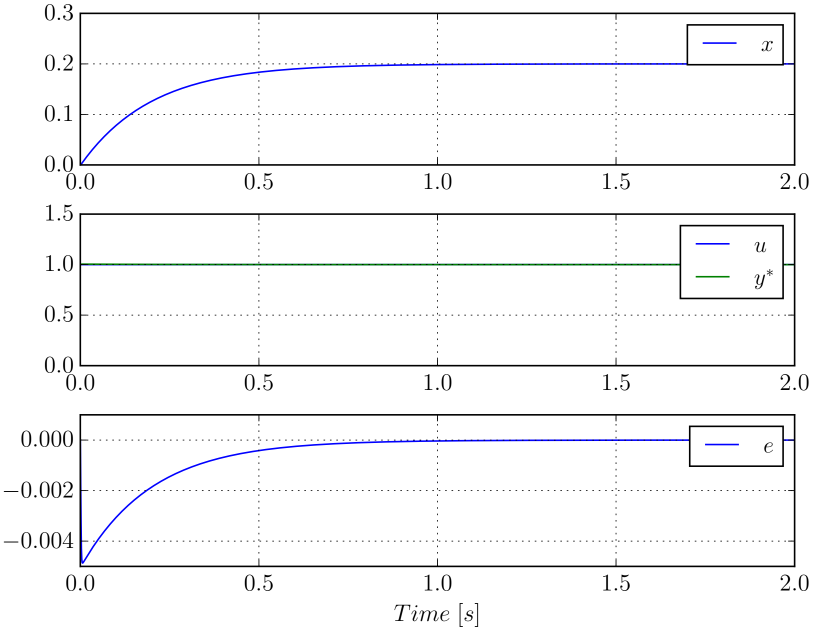


Figure 4.5: Convergence of $y^*(t)$ to u in Example 4.5

(4.17)*, and a plot of the system's algebraic solution is given in Figure 4.5 where an error between the two causality models are given as* $e = u - y^*(t)$*.*

As can be seen in the figure, $x(t)$ *and the error* e *seems to converge in about one second. Even though the initial condition for* $\xi(t)$ *was set in accordance with* (4.17) *there is a small error* $e \approx -5 \cdot 10^{-3}$ *at the start of the simulation, which converges to zero after about one second. Hence, this is an initial error which in this case is less than* 0.5% *of* u*.* ◇

Example 4.6 (System Simulation)**.** *The same system as given in Example 4.5 is to be simulated using the Euler integration method. First, the solver time-step size is set to* $\Delta t = 0.1T$ *and the results are shown in figure 4.6 where the error is defined as* $e = u - y^*(t)$*.*

As can be seen in the figure the maximal error is now about 10% *of* u*, which is a poorer result. This means that the solver error highly affects the transfer function based differentiation of* x *and by lowering the solver time step the error would decrease, and when* $\Delta t \to 0$ *the error would converge to the results shown in Example 4.5.* ◇

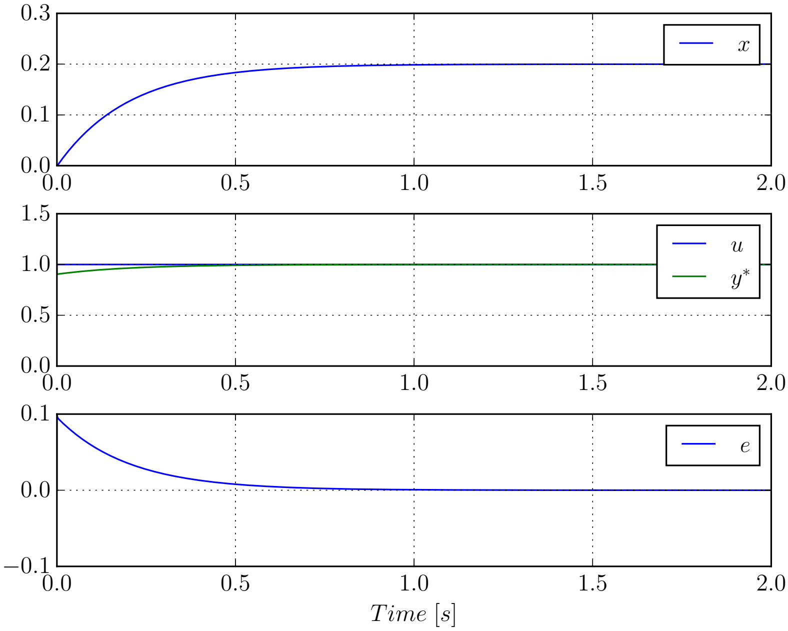


Figure 4.6: Convergence of $y^*(t)$ to u in Example 4.6

Example 4.7 (Varying Input)**.** *The same system as used in Example 4.5 and 4.6 is to be excited with*

$$u(t) = \sin(\omega t) \tag{4.29}$$

where $\omega = 2\pi$. A short solution time is wanted, thus, the band limitation time constant and solver step is chosen as $T = 0.01$ and $\Delta t = 0.01T$, respectively. Note that $T < \frac{2\pi}{\omega}$. The simulation results are shown in Figure 4.7. As can be seen in the figure, the maximal error is less than 4.9% of the amplitude of the variable input. ◇

The previous examples show that it is possible to choose T such that the error $e = u - y^*$ remains small. Also, it was shown that the solver error amplifies the error considerably in the estimate of $\dot{x}$ if the local solver time step is set high. The convergence of the error was algebraically proved for constant inputs, $\omega = 0$, when the solution was given algebraically. The reason why the error does not converge completely in Example 4.7 can be explained by looking at the low-pass filtering properties in the transfer function where the magnitude and the phase angle of the

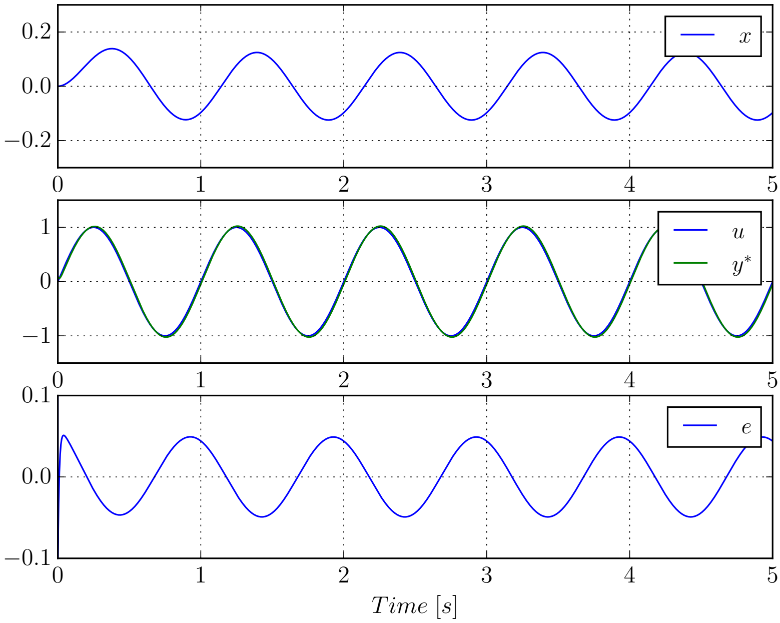


Figure 4.7: Convergence of $y^*(t)$ to u in Example 4.7

low-pass filter are expressed as

$$|H(j\omega)| = \frac{1}{\sqrt{T^2\omega^2+1}} \tag{4.30}$$

and

$$\angle H(j\omega) = -\arctan(T\omega) \tag{4.31}$$

respectively, where ω is the frequency of the input signal u^*. By assuming that the low-pass filter is excited with an input $u^* = \sin(\omega t)$, the output from the filter is given as

$$y = |H(j\omega)|\sin(\omega t + \angle H(j\omega)) \tag{4.32}$$

However, in Example 4.7 the low-pass filter is excited by $u^* = \dot{x} = h(ax+bu)$ where $u = \sin(\omega t)$. It is then possible to relate the input u to the output y^* in a transfer function, including the signal dynamics of u^* and the low-pass filter, and is given as

$$G(s) = \frac{y^*}{u}(s) = \frac{\frac{1}{T}(1-Ta)s - \frac{a}{T}}{s^2 + (\frac{1}{T}-a)s - \frac{a}{T}} \tag{4.33}$$

By using the same values as in the example it can be shown that $|G(j\omega)| = 1.029$ and $\angle G(j\omega) = -2.24°$ resulting in a maximal error of

$$\max\left[|G(j\omega)|\sin(\omega t + \angle G(j\omega)) - \sin(\omega t)\right] = 0.049 \tag{4.34}$$

which is 4.9% of the amplitude of u, the same as found in Example 4.7. Hence, the error introduced when estimating the derivative of a state using (4.13) is related to the low-pass filter magnitude and corresponding phase angle.

Based on these results it is reasonable to assume that values for T and Δt can be chosen such that the results is within a reasonable preset error tolerance, that, at least, is as good as solving the originally differential causality problem shown in Figure 4.3. Hence, the transfer function given in (4.13) is a good candidate for estimating $\dot{x}$ when fast solvable and explicit solutions of differential algebraic equations are considered and where algebraic solutions are not possible.

The transfer function based reformulation of the differential algebraic equations also enables the use of common stability analysis tools as well as making the formulation of hybrid causality models easier. As a matter of fact, the hybrid causality formulation has already been derived through the use of the transfer function, and the only thing left is to put it all together, as will be shown in the following.

4.4 Hybrid Causality Models

When working with systems that have the possibility to change causality or dynamics online during a simulation, hybrid causality would be a nice model feature, as discussed in Example 4.3. In section 4.3 a transfer function was used to reformulate differential algebraic equations into ordinary differential equations both in order to regain the lost state such that the number of states becomes constant and independent of the causality orientation, as well as being able to analyse model stability and using common numerical solvers for solving the system in an explicit manner. It also enables models to have multiple connectivity options, making them more modular, which would be a good model feature, at least in distributed systems.

Based on the reformulated differential causality in section 4.3.2, a hybrid causality model frame work can be established and expressed based on (4.8) and (4.16) as

$$\left.\begin{cases} \dot{x} = ax + bu \\ y = hx \end{cases}\right\} \quad q = 0$$

$$\left.\begin{cases} \dot{\xi} = -\dfrac{1}{T}\xi + \dfrac{1}{Th}u^* \\ y^* = -\dfrac{1}{Tb}\xi + \dfrac{1}{bh}\left(\dfrac{1}{T} - a\right)u^* \end{cases}\right\} \quad q = 1 \tag{4.35}$$

where q denotes the model form, $q = 0$ refers to integral causality and $q = 1$ refers to reformulated differential causality. Note that u and u^* are given from the model environment and y and y^* are the corresponding model outputs. To also enable non-linear dynamics, the hybrid causality model can be formulated as

$$\left.\begin{cases} \dot{x} = f(x, u, \tau_c) \\ y = h(x, u, \tau_c) \end{cases}\right\} \quad q = 0$$

$$\left.\begin{cases} x = h_x^{-1}(u^*, y^*, \tau_c) \\ \dot{\xi} = -\dfrac{1}{T}\xi + \dfrac{1}{T}x \\ z = -\dfrac{1}{T}\xi + \dfrac{1}{T}x \\ y^* = f_u^{-1}(z, x, \tau_c) \end{cases}\right\} \quad q = 1 \tag{4.36}$$

where $h_x^{-1}(\cdot)$ is the inverse of $h(\cdot)$ with respect to x and $f_u^{-1}(\cdot)$ is the inverse of $f(\cdot)$ with respect to $u = y^*$, where both the inverse functions are assumed to be unique. The only logics that need to be implemented in this type of model are related to model switching and resetting of integrators, which can be done in various and generic ways.

To illustrate the use of hybrid causality models, a marine power plant including hybrid causality models is thoroughly studied in a co-simulation in Example 4.8.

Example 4.8 (Marine Power Plant with Weak Power Grid)**.** *Consider the power plant given in Figure 4.8 including two generator sets (gensets), two circuit breakers with negligible dynamics and a power grid load. The gensets can either be connected or disconnected to the power grid, which is weak, meaning that a small disturbance in the power grid load would cause a disturbance in the active gensets. In other words, there is no capacitive effects in the power grid itself. Hence, one, but only one, genset at a time must set the power grid voltage. This means that when two or more gensets are connected to the same power grid, one genset must have a causality that outputs voltages while the others contribute with currents.*

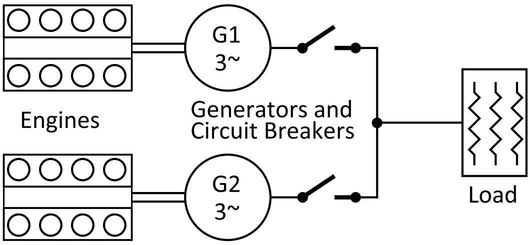


Figure 4.8: Marine power plant including two generator sets, two circuit breakers and a power grid load.

The integral causality model for a synchronous generator can according to [39] be expressed in the $(d, q, 0)$-reference frame as

$$\dot{\boldsymbol{\psi}} = -\omega_m \boldsymbol{D}\boldsymbol{\psi} - \boldsymbol{R}\boldsymbol{i} + \boldsymbol{E}\boldsymbol{u_{dq}} + \boldsymbol{b}u_f \tag{4.37a}$$

$$\boldsymbol{i} = \boldsymbol{L}^{-1}\boldsymbol{\psi} \tag{4.37b}$$

where ω_m is the engine speed, $\boldsymbol{\psi} = [\psi_d, \psi_q, \psi_f, \psi_D, \psi_Q]^\top$ is the magnetic fluxes for d, q, the field and the damping in d and q, respectively, $\boldsymbol{i} = [i_d, i_q, i_f, i_D, i_Q]^\top$ is the current vector, $\boldsymbol{u}_{dq} = [u_d, u_q]^\top$ is the voltage vector containing the voltages for d and q, u_f is the field voltage that controls the power grid rms voltage and

$$\boldsymbol{D} = \begin{bmatrix} 0 & -n_p & 0 & 0 & 0 \\ n_p & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \ \boldsymbol{E} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4.38}$$

$$\boldsymbol{R} = \begin{bmatrix} R_d & 0 & 0 & 0 & 0 \\ 0 & R_q & 0 & 0 & 0 \\ 0 & 0 & R_f & 0 & 0 \\ 0 & 0 & 0 & R_D & 0 \\ 0 & 0 & 0 & 0 & R_Q \end{bmatrix}, \ \boldsymbol{b} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{4.39}$$

$$\boldsymbol{L} = \begin{bmatrix} L_d & 0 & L_{df} & L_{dD} & 0 \\ 0 & L_q & 0 & 0 & L_{qQ} \\ L_{df} & 0 & L_f & L_{fD} & 0 \\ L_{dD} & 0 & L_{fD} & L_D & 0 \\ 0 & L_{qQ} & 0 & 0 & L_Q \end{bmatrix} \tag{4.40}$$

Here, $\boldsymbol{R}$ is the internal resistance matrix, $\boldsymbol{L}$ is the inductance matrix and n_p is the number of pole pairs in the generator. The model outputs are the currents i_d and i_q.

Correspondingly, the equations for the generator model that outputs the voltages u_d *and* u_q *can be expressed as*

$$\boldsymbol{u}_{dq} = \dot{\boldsymbol{\psi}}_{dq} + \omega_m \boldsymbol{D}_{dq} \boldsymbol{\psi}_{dq} + \boldsymbol{R}_{dq} \boldsymbol{i}_{dq} \tag{4.41a}$$

$$\dot{\boldsymbol{\psi}}_{fDQ} = -\boldsymbol{R}_{fDQ} \boldsymbol{i}_{fDQ} + \boldsymbol{b}_{fDQ} u_f \tag{4.41b}$$

where $\boldsymbol{u}_{dq} = [u_d, u_q]^\top$, $\boldsymbol{\psi}_{dq} = [\psi_d, \psi_q]^\top$, $\boldsymbol{i}_{dq} = [i_d, i_q]^\top$, $\boldsymbol{\psi}_{fDQ} = [\psi_f, \psi_D, \psi_Q]^\top$, $\boldsymbol{i}_{fDQ} = [i_f, i_D, i_Q]^\top$ *and*

$$\begin{aligned} \boldsymbol{R}_{dq} &= \begin{bmatrix} R_d & 0 \\ 0 & R_q \end{bmatrix}, \ \boldsymbol{D}_{dq} = \begin{bmatrix} 0 & -n_p \\ n_p & 0 \end{bmatrix} \\ \boldsymbol{R}_{fDQ} &= \begin{bmatrix} R_f & 0 & 0 \\ 0 & R_D & 0 \\ 0 & 0 & R_Q \end{bmatrix}, \ \boldsymbol{b}_{fDQ} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \end{aligned} \tag{4.42}$$

It is then possible to rearrange (4.37b) *such that*

$$\begin{bmatrix} \boldsymbol{\psi}_{dq} \\ \boldsymbol{i}_{fDQ} \end{bmatrix} = \boldsymbol{Z} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{4.43}$$

where $\boldsymbol{Z}$ *is given as*

$$\boldsymbol{Z} = \begin{bmatrix} Z_{11} & 0 & \frac{L_{dD}L_{fD}-L_D L_{df}}{L_{fD}^2-L_D L_f} & \frac{L_{fD}L_{df}-L_{dD}L_f}{L_{fD}^2-L_D L_f} & 0 \\ 0 & -\frac{L_{qQ}^2-L_Q L_q}{L_Q} & 0 & 0 & \frac{L_{qQ}}{L_Q} \\ -\frac{L_{dD}L_{fD}-L_D L_{df}}{L_{fD}^2-L_D L_f} & 0 & -\frac{L_D}{L_{fD}^2-L_D L_f} & \frac{L_{fD}}{L_{fD}^2-L_D L_f} & 0 \\ -\frac{L_{fD}L_{df}-L_{dD}L_f}{L_{fD}^2-L_D L_f} & 0 & \frac{L_{fD}}{L_{fD}^2-L_D L_f} & -\frac{L_f}{L_{fD}^2-L_D L_f} & 0 \\ 0 & -\frac{L_{qQ}}{L_Q} & 0 & 0 & \frac{1}{L_Q} \end{bmatrix} \tag{4.44}$$

where

$$Z_{11} = \frac{L_f L_{dD}^2 - 2L_{dD}L_{fD}L_{df} + L_d L_{fD}^2 + L_D L_{df}^2 - L_D L_d L_f}{L_{fD}^2 - L_D L_f} \tag{4.45}$$

This means that ψ_{dq} *can be calculated from* (4.43)*, differentiated and inserted into* (4.41a) *to obtain* $\boldsymbol{u}_{dq}$*. The current vector* $\boldsymbol{i}_{fDQ}$ *is found by first obtaining* $\dot{\boldsymbol{\psi}}_{fDQ}$ *from integrating* (4.41b) *and inserting into* (4.43).

In this case the differential causality model contains only three states while the integral causality model contains five. However, based on the transfer function-based differentiation discussed in section 4.3.2, the differential algebraic equations *in* (4.41a) *can be reformulated to differential equations. By defining*

$$\boldsymbol{\psi}_{dq} = \boldsymbol{Z}_{dq} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{4.46}$$

where $\boldsymbol{Z}_{dq} \in \mathcal{R}^{2\times 5}$ *contains the two first rows in* $\boldsymbol{Z}$ *and that* $\dot{\hat{\boldsymbol{\psi}}}_{dq}$ *is the approximated derivative of* $\boldsymbol{\psi}_{dq}$ *obtained from the transfer function, the reformulated differential causality model can be expressed as*

$$\dot{\boldsymbol{\psi}}_{fDQ} = \boldsymbol{b}_{fDQ} u_f - \boldsymbol{R}_{fDQ} \boldsymbol{i}_{fDQ} \tag{4.47a}$$

$$\dot{\boldsymbol{\xi}}_{dq} = -\frac{1}{T}\boldsymbol{\xi}_{dq} + \frac{1}{T}\boldsymbol{Z}_{dq} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{4.47b}$$

$$\boldsymbol{i}_{fDQ} = \boldsymbol{Z}_{fDQ} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \tag{4.47c}$$

$$\begin{aligned} \boldsymbol{u}_{dq} &= -\frac{1}{T}\boldsymbol{\xi}_{dq} + \boldsymbol{R}_{dq}\boldsymbol{i}_{dq} \\ &\quad + \left(\frac{1}{T}\boldsymbol{I}_{2\times 2} + \omega_m \boldsymbol{D}_{dq}\right) \boldsymbol{Z}_{dq} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \end{aligned} \tag{4.47d}$$

where $\boldsymbol{Z}_{fDQ} \in \mathcal{R}^{3\times 5}$ *contains the three last rows in* $\boldsymbol{Z}$ *given in* (4.44) *and* $\boldsymbol{I}_{2\times 2}$ *is the identity matrix of size 2.*

The reformulated differential causality model can also be expressed more compactly as

$$\begin{bmatrix} \dot{\boldsymbol{\psi}}_{fDQ} \\ \dot{\boldsymbol{\xi}}_{dq} \end{bmatrix} = \boldsymbol{A}_{\psi,\xi} \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix} + \boldsymbol{B}_{\psi,\xi}\boldsymbol{i}_{dq} + \boldsymbol{b}_{\psi,\xi} u_f \tag{4.48a}$$

$$\begin{aligned} \boldsymbol{u}_{dq} &= -\frac{1}{T}\boldsymbol{\xi}_{dq} + \boldsymbol{R}_{dq}\boldsymbol{i}_{dq} \\ &\quad + \left(\frac{1}{T}\boldsymbol{I}_{2\times 2} + \omega_m \boldsymbol{D}_{dq}\right) \boldsymbol{Z}_{dq} \begin{bmatrix} \boldsymbol{i}_{dq} \\ \boldsymbol{\psi}_{fDQ} \end{bmatrix} \end{aligned} \tag{4.48b}$$

where

$$\begin{aligned} \boldsymbol{A}_{\psi,\xi} &= \begin{bmatrix} -\boldsymbol{R}_{fDQ} \left\{ \begin{matrix} Z_{33} & Z_{34} & Z_{35} \\ Z_{43} & Z_{44} & Z_{45} \\ Z_{53} & Z_{54} & Z_{55} \end{matrix} \right\} & \boldsymbol{0}_{3\times 2} \\ \frac{1}{T} \left\{ \begin{matrix} Z_{13} & Z_{14} & Z_{15} \\ Z_{23} & Z_{24} & Z_{25} \end{matrix} \right\} & -\frac{1}{T}\boldsymbol{I}_{2\times 2} \end{bmatrix} \\ \boldsymbol{B}_{\psi,\xi} &= \begin{bmatrix} -\boldsymbol{R}_{fDQ} \left\{ \begin{matrix} Z_{31} & Z_{32} \\ Z_{41} & Z_{42} \\ Z_{51} & Z_{52} \end{matrix} \right\} \\ \frac{1}{T} \left\{ \begin{matrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{matrix} \right\} \end{bmatrix}, \quad \boldsymbol{b}_{\psi,\xi} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{aligned} \tag{4.49}$$

and where Z_{ij} is the element $\boldsymbol{Z}(i, j)$ in (4.44).

In order to enable smooth switching between the two causality orientations online during a simulation, initial conditions must be derived. When switching from the integral causality model to the reformulated differential causality model, the initial conditions for the simulation time t_0 for the states in the reformulated differential causality model are given as

$$\begin{aligned}\boldsymbol{\psi}^{rD}_{fDQ}(t_0) &= \boldsymbol{\psi}^{I}_{fDQ}(t_0)\\ \boldsymbol{\xi}_{dq}(t_0) &= -T\boldsymbol{u}^{I}_{dq}(t_0) + T\boldsymbol{R}_{dq}\boldsymbol{i}^{I}_{dq}(t_0)\\ &+ \left(\boldsymbol{I}_{2\times 2} + T\omega_m(t_0)\boldsymbol{D}_{dq}\right)\boldsymbol{Z}_{dq}\begin{bmatrix}\boldsymbol{i}^{I}_{dq}(t_0)\\ \boldsymbol{\psi}^{I}_{fDQ}(t_0)\end{bmatrix}\end{aligned} \tag{4.50}$$

Note that the superscripts ^{rD} and ^{I} are abbreviations for the variables in the reformulated differential causality model and the integral causality model, respectively, in order to separate equal variable names. Correspondingly, the initial conditions when switching from the reformulated differential causality model to the integral causality model at simulation time t_0 are given as

$$\begin{aligned}\boldsymbol{\psi}^{I}_{fDQ}(t_0) &= \boldsymbol{\psi}^{rD}_{fDQ}(t_0)\\ \boldsymbol{\psi}^{I}_{dq}(t_0) &= \boldsymbol{Z}_{dq}\begin{bmatrix}\boldsymbol{i}^{rD}_{dq}(t_0)\\ \boldsymbol{\psi}^{rD}_{fDQ}(t_0)\end{bmatrix}\end{aligned} \tag{4.51}$$

To test the hybrid causality model, the total system shown in Figure 4.8 has been implemented in the Python programming language as a co-simulation where each genset and all controls system are treated as separate submodels. The field voltages u_f for the two generators are controlled by PI-controllers, where keeping the desired rms voltage is the control objectives in addition to control reactive load sharing. The PI control laws are saturated such that $u_f \in [0.0, 100.0]V$. The two generators are powered by auxiliary diesel engines. These auxiliary engines are also controlled by PI controllers, acting as governors, and is also saturated according to realistic fuel injection limitations. A simple power management system (PMS) is also implemented in order to allow generator synchronization and load sharing properties. It is out of scope here to go into detail regarding the additional subsystems, but they have been thoroughly documented in [39], see Chapter 6. Also, the main parameters for the generator models are the same as used in [39] and are given in Table 4.1. Note that each genset has a rated capacity of 2438 kW.

In this case study the rms voltage reference is set to 690 V, the auxiliary engine speed reference to 600 RPM when started but in idle mode, and 720 RPM when being synchronizing or connected to the power grid, corresponding to a grid frequency of 60 Hz. In the simulation generator 1 is first started and the power grid

Table 4.1: Generator parameters

Parameter	Value	Parameter	Value
L_d	0.0007728 H	L_{qQ}	0.0104431 H
L_D	0.5987330 H	n_p	5 –
L_f	0.6063750 H	R_d	0.0049700 Ω
L_{dD}	0.0162176 H	R_D	6.2165657 Ω
L_{df}	0.0162176 H	R_q	0.0049700 Ω
L_{fD}	0.5769750 H	R_Q	9.7575356 Ω
L_q	0.0005257 H	R_f	0.3150000 Ω
L_Q	0.3987454 H	f_G	60 Hz

load is activated at t =25 s. Generator 2 is also started and set to run in idle mode before starting to synchronize at t =30 s. At t =110 s generator 1 is set to disconnect from the power grid and go into idle mode, which means that generator 2 must take all the load and set the power grid voltage.

The power grid load is set to 1000 kW in active load and 1000 kVAr in reactive load. In addition, noise is added to both the active- and the reactive power grid load in order to make the simulation even more realistic and to test the total power plant stability. The length of the simulation is set to 150 s, and the Euler integration method is used locally to solve each subsystem separately with a time step Δt =0.1 ms, while the global co-simulation time step is set to 0.2 ms, meaning that simulation data are exchanged every other local solver time step. The low-pass filter time constants in the reformulated differential causalities are set to 1.0 ms and the simulation results regarding the generator models are shown in Figure 4.9.

The upper leftmost plot in the figure shows the rms voltages for the two generators which seem to converge to the references rms voltage when neglecting the small oscillations due to the noisy power grid load. The simulation results given in the plot to the right show that the change in causality, as shown in the last plot in the figure and happens about 23 s after initiating synchronization of generator 2, does not affect the power grid voltage due to the new initial conditions. Note that when both generators are connected to the power grid the magnitude of the oscillations seems to decrease since both generators contribute to reducing the oscillations through their field controllers. The results also show that when the power grid load is activated, generator 1 gets a small reduction in the rms voltage but manages to restore it to the desired value in less than 10 s.

The second and the third leftmost plots show the produced active and reactive generator powers, respectively, in comparison to the active and reactive power consumed by the power grid load. As the simulation results show, when generator 2 has been connected to the power grid, the active and the reactive load is be-

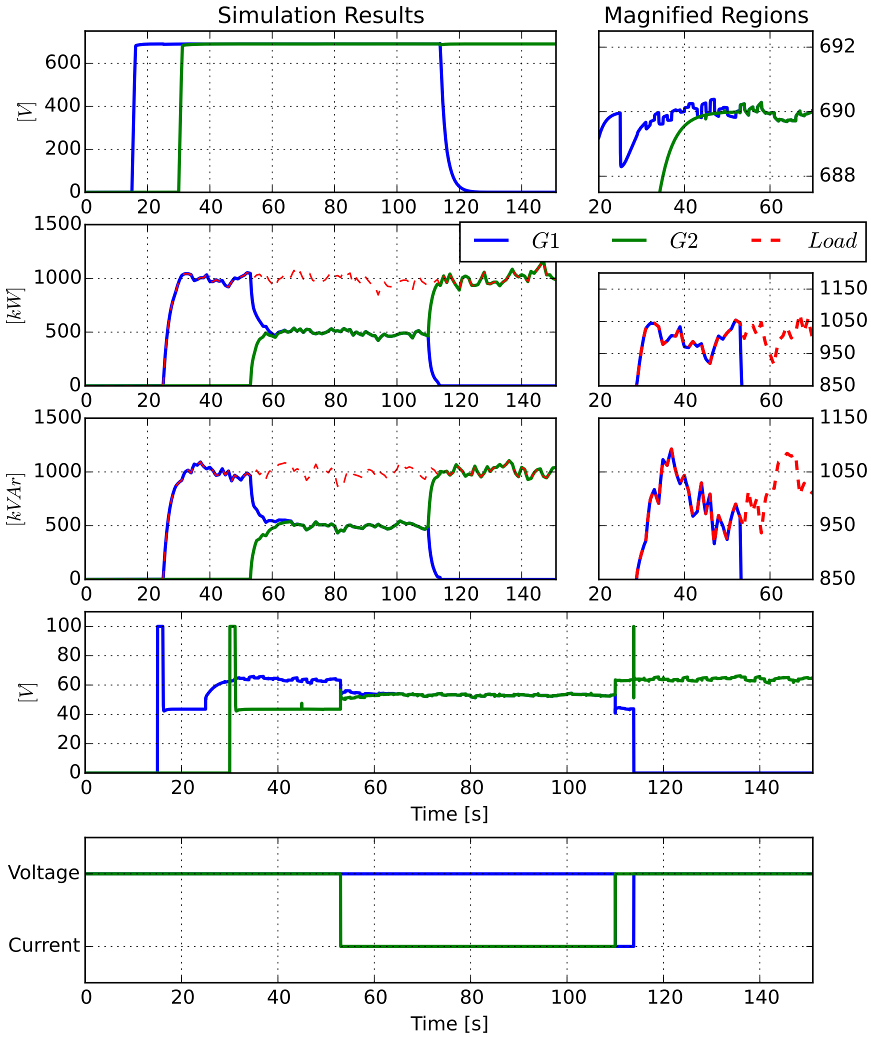


Figure 4.9: Simulation results for power plant with hybrid generator models

ing shared between the two generators equally. This means that the load sharing functionalities in the PMS work properly. The corresponding rightmost plots show magnified regions of the active and the reactive power grid load when generator 2 is connected. These results also indicate that the causality change of generator 2 does not affect the simulation results when having in mind that load sharing is *initiated right afterwards.*

The fourth plot shows the field voltages (u_f) for the two generators. Also these results seem to be stable during the entire simulation. The results also show that when the power grid load is activated the field voltage for generator 1 increases, and when generator 2 is connected to the grid the two field voltages converge due to equal load sharing. The last plot shows the causality configurations for the two generator models during the entire simulation. The results show that generator 2 is demanded by the PMS to set the power grid voltage before generator 1 is being disconnected from the power grid at about t =120 s.

Figure 4.10 shows the simulation results for the two auxiliary diesel engines. The upper leftmost plot in the figure shows the speeds of the two engines which seems to converge to 720 RPM when the generators are connected to the power grid. A magnified region of the engine speeds are shown to the right when generator 2 is synchronized and connected to the grid. As can be seen in the magnified plot, generator 2 has a lower speed than generator 1 right before being connected to the power grid. This has to do with phase synchronization between the two generators.

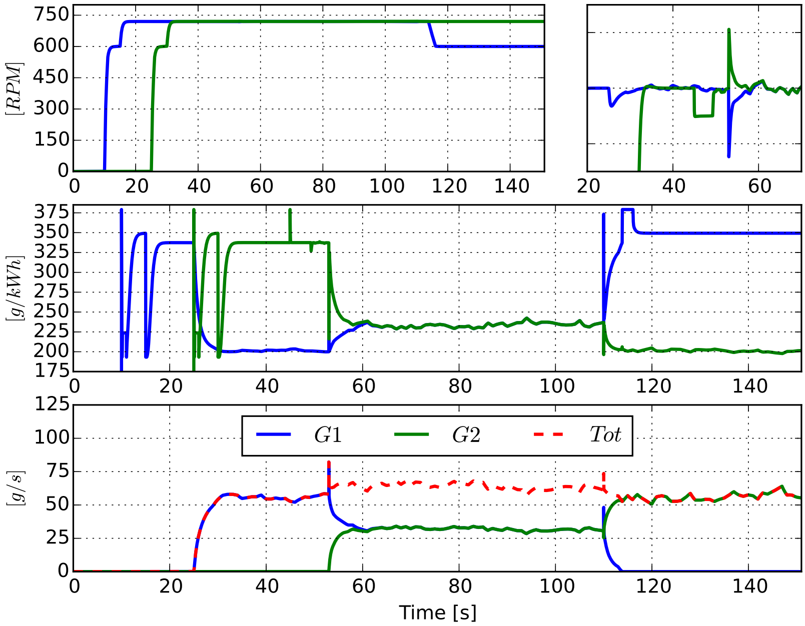


Figure 4.10: Simulation results for auxiliary diesel engines driving the generators

The results also show that both generators get opposite peaks when generator 2 is connected. This has nothing to do with changing the causality of generator 2, but due to active load sharing functionalities initiated by the PMS.

The second plot in the figure shows the brake specific fuel consumption (BSFC) for the two auxiliary diesel engines. The results show that when only one generator is active the BSFC is lower than when both generators are active. This has to do with the efficiency of the engines, which is highest when the engines approach a loading of about 90 %. When both generators are connected they have about 25 % loading each, which gives a higher BSFC. Also note that the BSFC for when an engine is in idle shown in the figure is not correct since the BSFC curve is only valid for engine loadings above 10 %. In reality the BSFC for such an engine in idle can be as high as 1700-2000 g/kWh, maybe more. The corresponding instantaneous fuel consumption is shown in the last plot in the figure. It is also possible to see the drop in efficiency in these results, and when both generators are connected the instantaneous fuel consumption increases with about 20 %.

This case study illustrates how hybrid causality models can be used in practice and shows that such models are well suited for co-simulation purposes where the I/O mapping may change during the simulation. The presented marine power plant model seems to work properly and gives a good foundation for further research of marine power plants and corresponding control- and optimization strategies. It is however out of scope to investigate this further here.

◇

In co-simulations, as well as in other simulations, model stability is important and a is discussed in the following.

4.5 Stability of Hybrid Causality Models

For models with reformulated differential causalities and static causality orientations, the traditional stability analysis tools such as Lyapunov stability theory can be applied directly. This, because such models do not longer contain differential algebraic equations and can be represented in full state-space formulations. Hence, the focus here will be devoted to dynamical stability for hybrid causality models. A hybrid modeling framework with corresponding stability theory is presented in [90]. The hybrid modeling framework is divided into two time domains, one propagating in continuous time and the other in discrete time. The hybrid causality model framework presented in (4.35) fits this hybrid framework where the two causality orientations propagate in continuous time and where the switching between the causality orientations constitute the discrete time events.

In general, the main stability theory regarding the hybrid framework in [90] states that a hybrid model is stable whenever it can be proved that the level of energy in a hybrid model decreases in both continuous time and discrete time. It also states that if the level of energy is constant in discrete time but decreases in continuous time, it is enough to prove that the model does not stay in the discrete time domain forever and that it ends up in the continuous time domain in order to assure stability. These stability results are useful also for hybrid causality models. Hence, if it can be proven that the level of energy in a hybrid causality model does not increase in discrete time, the stability conditions for hybrid causality models are easy to formulate. As a matter of fact, the initial conditions for the causality orientations in a hybrid causality model are strongly related to the level of energy for the model in discrete time. This is summarized in Lemma 4.1.

Lemma 4.1 (Constant Energy Level in Discrete Time)**.** *The level of energy in discrete time for a hybrid causality model is only dependent on the initial conditions for the causality orientations, and the level of energy in a hybrid causality model is constant when choosing the initial conditions for* (4.8) *and* (4.16) *to be energy conserving, such as in* (4.18) *and in* (4.17)*, respectively.*

Proof. Assume that the power exchange between a submodel and the model environment for the system given in (4.8) and (4.16) is given as

$$P = y \cdot u, \qquad P^* = y^* \cdot u^* \tag{4.52}$$

respectively, and that a power residual is defined as the difference between these two in discrete time, $\Delta P := P - P^*$. By including system dynamics, the power for the integral causality orientation and the reformulated differential causality orientation can be expressed as

$$\begin{aligned} P &= y \cdot u = hxu = u^* y^* \\ P^* &= y^* \cdot u^* = \left[-\frac{1}{Tb}\xi + \frac{1}{bh}\left(\frac{1}{T} - a\right) u^* \right] u^* \\ &= \left[-\frac{1}{Tb}(1 - aT)x + u + \frac{1}{b}\left(\frac{1}{T} - a\right) x \right] hx \\ &= uy \end{aligned} \tag{4.53}$$

which tell that $\Delta P = 0$ during causality switching for hybrid causality models as expressed in (4.35). This means that power is neither dissipated nor generated during causality switching. □

Based on Lemma 4.1 the stability conditions for hybrid causality models can be formulated as in Theorem 4.1.

Theorem 4.1 (Dynamical Stability Conditions for Hybrid Causality Models)**.** *For a given hybrid causality model containing multiple causality forms assume that the differential causalities have been reformulated such that differential algebraic equations are changed to differential equations. Also, assume that the initial conditions for the causality orientations are chosen such that no energy gets dissipated or generated when switching causality orientation, and that the hybrid causality model cannot stay in discrete time forever, having a finite number of causality switchings. Then, the dynamical stability for each causality orientation can be analysed separately with suited stability theories in order to assure total dynamical stability for the model.*

Proof. Based in Lemma 4.1, the level of energy for a hybrid causality model is constant in discrete time if the initial conditions are energy conservative. Hence, it follows directly that if all causality orientations dissipate more energy than it produces and that the model propagates more in continuous time than discrete time, and more importantly ends up in continuous time, the level of energy in the total model will decrease, which implies stability. □

To illustrate the usage of Theorem 4.1, the stability of the hybrid generator model presented in Example 4.8 is presented in Example 4.9.

Example 4.9 (Stability Analysis of Hybrid Generator Model)**.** *Consider the hybrid causality generator model presented in Example 4.8. Starting with the complete integral causality model, as expressed in* (4.37)*, a Lyapunov function candidate is given as*

$$V_i = \frac{1}{2}\boldsymbol{\psi}^\top \boldsymbol{I}\boldsymbol{\psi} \tag{4.54}$$

where $\boldsymbol{I}$ is the identity matrix. Hence, it can be verified that

$$\dot{V}_i = \boldsymbol{\psi}_i^\top \left(-\omega_m \boldsymbol{D} - \boldsymbol{R}\boldsymbol{L} - \boldsymbol{E}\boldsymbol{R}_L \boldsymbol{J}_0 \boldsymbol{L}^{-1}\right)\boldsymbol{\psi} + \boldsymbol{\psi}^\top \boldsymbol{b} u_f \tag{4.55}$$

where $\boldsymbol{R}_L$ is the external load resistance matrix and $\boldsymbol{J}$ is a mapping matrix such that $\boldsymbol{i}_{dq} = \boldsymbol{J}_0 \boldsymbol{i}$. For simplicity, assume that the PID-controller controlling the field voltage is stable. Then, it can be verified that

$$\dot{\hat{V}}_i \leqslant \boldsymbol{\psi}_i^\top \left(-\omega_m \boldsymbol{D} - \boldsymbol{R}\boldsymbol{L} - \boldsymbol{E}\boldsymbol{R}_L \boldsymbol{J}_0 \boldsymbol{L}^{-1}\right)\boldsymbol{\psi} \leqslant 0 \tag{4.56}$$

for all energy dissipative power grid loads $\boldsymbol{R}_L$ and all generator speeds ω_m given the parameters in Table 4.1, where $\dot{V}_i = \dot{\hat{V}}_i + \boldsymbol{\psi}^\top \boldsymbol{b} u_f$. Correspondingly, consider *the reformulated differential causality orientation given in* (4.48)*. A Lyapunov function candidate is given as*

$$V_{rD} = \frac{1}{2} \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix}^{\top} \boldsymbol{I} \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix} \tag{4.57}$$

It can be verified that

$$\begin{aligned} \dot{V}_{rD} = & \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix}^{\top} \left(\boldsymbol{A}_{\psi,\xi} - \boldsymbol{B}_{\psi,\xi}\boldsymbol{R}_L^{-1}\boldsymbol{J}_1\right) \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix} \\ & + \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix}^{\top} \boldsymbol{b}_{\psi,\xi} u_f \end{aligned} \tag{4.58}$$

where $\boldsymbol{J}$ *is a mapping matrix such that*

$$\boldsymbol{u}_{dq} = \boldsymbol{J}_1 \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix} \tag{4.59}$$

By also here assuming that the PID-controller controlling the field voltage is stable, it can be verified that

$$\dot{\hat{V}}_{rD} \leqslant \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix}^{\top} \left(\boldsymbol{A}_{\psi,\xi} - \boldsymbol{B}_{\psi,\xi}\boldsymbol{R}_L^{-1}\boldsymbol{J}_1\right) \begin{bmatrix} \boldsymbol{\psi}_{fDQ} \\ \boldsymbol{\xi}_{dq} \end{bmatrix} \leqslant 0 \tag{4.60}$$

for all energy dissipative power grid loads $\boldsymbol{R}_L$ *and all generator speeds* ω_m *given the parameters in Table 4.1, where*

$$\dot{V}_{rD} = \dot{\hat{V}}_{rD} + \begin{bmatrix} \boldsymbol{\psi}_{fDQ} & \boldsymbol{\xi}_{dq} \end{bmatrix} \boldsymbol{b}_{\psi,\xi} u_f \tag{4.61}$$

Hence, the total hybrid causality generator model is dynamically stable. ◇

When it comes to stability in distributed systems, it is in most cases more interesting to look at simulator stability rather than dynamical stability since simulator stability contains both dynamical stability and numerical stability. However, this is out of the scope in this chapter, but has been treated thoroughly in [34, 36] and in Chapter 3.

4.6 Conclusion

In this chapter mathematical models of physical systems with multiple causality configurations and connectivity options, formulated as hybrid causality models, have been studied, as a major step in solving the issues related to systems that are tightly coupled through causality. The causality configurations that imply differential algebraic equations has been reformulated to ordinary differential equations by the use of transfer function based differentiation. This, mainly in order to retain a full state space model, consisting of only ordinary differential equations, for stability analysis purposes and to facilitate the use of simple solvers such as the Euler integration method.

A convergence study of the transfer function based differentiation were initiated and the results showed that the transfer function based differentiation method works well as long as the differentiation time constant is chosen such that the filter dynamics are faster than the dynamics of the physical system, as well as keeping all eigenvalues inside the stability region of the chosen numerical solver.

Various examples have been given to facilitate readability and to highlight some of the ideas presented in this chapter. A more in-depth example presenting a marine power plant is given to show applicability and to highlight practical usage of the proposed theory. This marine power plant is to be further elaborated in Chapter 6.

Part II

Modeling and Control

All models are wrong, but some are useful!
- George E. P. Box (1919-2013)

CHAPTER 5

Modeling of Generic Offshore Vessel including Crane

This chapter is based on a restructured version of [38, see **P6** in section 1.6] where a generic marine offshore vessel model including a deck crane with three degrees of freedom is derived using Lagrange's method. A framework for deriving such rigid body dynamics, including quasi coordinates, based on power bond connections is also presented. This method for deriving the state-space model for two systems tightly coupled through causality (see chapter 4) is exact, in comparison to the method of reformulating differential algebraic equations into differential equations, as presented in section 4.3. A case study of a marine offshore vessel in DP-operation is conducted and simulations are performed. However, note that no co-simulations are conducted in this chapter. Later on, in chapter 8 the total case study system presented here will be divided into subsystems and simulated in a co-simulation where the vessel and the crane are packed as one subsystem. This chapter also shows that even small payloads, in comparison to the vessel itself, attached to crane through a wire [125] has a considerable impact on the performance of the vessel.

5.1 Introduction

A number of maritime simulator solutions have previously been developed for different applications. Two examples are the *Marine Systems Simulator*, *MSS* [126, 127], and the *Marine Vessel and Power Plant Simulator* [6, 128, 129]. The MSS simulator, a joining together of the GNC-toolbox [130], MCsim, [131], and DCMV [132], is an environment or a platform which allows for rapid formulation of dynamic equations for vessel, with special focus on maritime control systems. This framework is implemented in the Matlab/Simulink^{R} software. However, it does not facilitate crane and manipulator extensions. The Marine Vessel and Power

Plant Simulator is a Matlab/Simulink^{R}-based extension of the MSS-simulator, that supports better thruster models and electric power plants. The main purpose of the simulator is to support development of advanced power system control and optimization methods [129]. As noted in [129], the disadvantage of using Simulink is that it is hard to model interconnections. It is noted that the system is hard to divide into the levels as required by the subsystem architecture of Simulink. Bond graph theory, on the other hand, provides a unified description of physical systems across multiple energy domains [133], something that, in turn, makes interconnection of subsystems convenient.

Three references that does support interconnected crane and manipulator dynamics are [134], [135] and [136]. The former reference derives dynamic equations for AUVs using an Lagrangian approach, similar to what is done here. The main purposes of this approach in [134] is (i) to avoid the singularities that arise when using an Euler-angle representation, while at the same time keeping a minimal formulation (i.e., not using e.g. a unit quaternion-representation), by instead using quasi-coordinates, and (ii), to enable AUV-manipulator modelling. The main difference between this approach, and the approach presented here, is that our mathematical model is formulated as shown in [25] and [137, 138], rather than in the traditional manner following the Lagrangian formulation as presented in e.g. [139, 24], thus enabling us to represent the model in bond graphs. In addition to allowing for a bond graph representation, this is also advantageous because we avoid the task of time-differentiating the mass-inertia matrix. The second reference, [135], presents a modeling approach for heavy marine lifting operations based on the modeling and simulation software 20-sim, using the 20-sim 3D Mechanics toolbox [140]. This approach provides a bond graph interface to other subsystems. It is however a disadvantage that the approach is constrained to a particular software implementation and a particular toolbox within the software, something that in turn provides a number of limitations and restrictions. One example is that the 3D Mechanics toolbox will only allow for diagonal mass-inertia matrices and linear spring and damper relations. Our approach, on the other hand is not limited to any particular software implementation. Rather, the model is based on general bond graph theory, and can, as such be implemented in any software that supports scripting, since the system equations can easily be extracted from the bond graphs. However, it is convenient to use a software that directly supports bond graphs, as will be done here. The third reference, [136], does not depend on particular software, but is instead based on stiff spring connections between rigid bodies. This is done in order to resolve problems related to derivative causality appearing when connecting the rigid bodies. The disadvantage with this approach is that if a soft spring is used, the accuracy will be severely affected, while if a stiff spring is used, fast dynamics that will increase the simulation time will appear. As such, a compromise between slow simulations and large simulation errors must be made when deciding on the spring stiffness. In the framework presented in this chapter, the derivative causalities have been resolved algebraically, and no compliance between rigid bodies are necessary. The advantage with using the compliance-approach in [136] is that it is easier to alter the structure of the crane, for example, by replacing a revolute crane joint with a linear one.

Our contribution then is to provide a framework that allows for effective simulator development based on the bond graph methodology, something which makes it well suited for multi energy-domain modeling and also arguably better suited for representing physical systems than, for example, block-diagrams [31], as seen both from a practical modelling-technical point of view, and from a pedagogical point of view [141]. The former is something we see as highly relevant for maritime vehicles as they, at least, will include components from the mechanical, electrical and hydraulic domain. Currently, there are a number of simulators specialized for various purposes, as pointed out earlier, but they lack the flexibility associated to bond graph modelling in terms of generic interfaces to subsystems, does not facilitate connection of cranes with true dynamic interconnections to the vessel, or are limited in terms of rigid body dynamics because they are is based on very specialized toolboxes, (i.e., the 3D Mechanics toolbox for [135]). The framework presented in the following, on the other hand, retain all the advantages of bond graphs, while remaining independent of any particular software or toolboxes, and has no limitations or restrictions regarding the rigid body dynamics. As such, this is the first software independent bond graph framework for crane and vessel dynamics where the interconnections are modeled according to true physical rigid body principles without non-physical limitations such as diagonal mass-inertia matrix.

5.2 Marine Vehicle Dynamics

In this section we seek to find equations of motion for the marine vehicle, using momentum and displacement as generalized states. A state space model expressed in terms of these states is convenient for bond graph implementation, as will be seen later on. Besides, this state space model is far easier to derive from the Lagrangian equations than the traditional state space model for marine vehicles persented in e.g. [14], with displacements and displacement rates as states. This is mainly because we avoid the tedious task of time differentiating the mass-inertia matrix when using momentum, as opposed to displacement rates. Figure 5.1 can be used as a reference for some of the variables in this section.

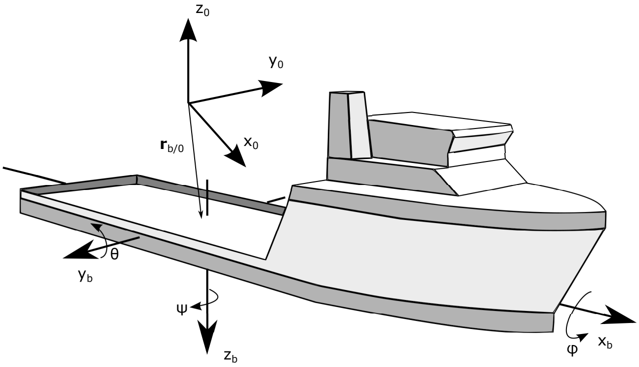


Figure 5.1: Illustration of kinematics of a marine vehicle.

5.2.1 Kinematic Relations

Let the position and orientation of the vehicle be given relative to an inertial reference frame, denoted by 0. Attach a second reference frame to the vehicle body and denote it b. The position of the vehicle is given by the vector $\boldsymbol{r}^0_{b/0}$, where the superscript indicate that the vector is expressed in terms of the inertial reference frame, while the subscript $b/0$ indicate that the vector gives the position to the origin of the vehicle body fixed reference frame relative to the origin of the inertial reference frame. The orientation of the the vehicle is given by the Euler angles $\boldsymbol{\Theta} = [\phi,\, \theta,\, \psi]^T$. In this chapter, the Euler angles are defined such that if the vehicle is rotated an angle ϕ about its x-axis, an angle θ about the resulting y-axis, and finally an angle ψ about the resulting z-axis, then the body fixed coordinate frame have the same orientation as the inertial reference frame. Using this, we can find an expression for the angular velocity of the vehicle, expressed in terms of the body fixed reference frame as

$$\boldsymbol{\omega}^b_{b/0} = \boldsymbol{i}_b\dot{\phi} + \boldsymbol{j}^{'}_b\dot{\theta} + \boldsymbol{k}^{''}_b\dot{\psi} = \boldsymbol{T}^{-1}_{\Theta}(\boldsymbol{\Theta})\dot{\boldsymbol{\Theta}} \tag{5.1}$$

where $\boldsymbol{i}_b$ is the unit normal vector along the x-axis of the vehicle body fixed reference frame, $\boldsymbol{j}^{'}_b$ is the unit normal vector along the y-axis of the reference frame resulting from the rotation ϕ, and $\boldsymbol{k}^{''}_b$ is the unit normal vector along the z-axis of the reference frame resulting from the rotation θ about $\boldsymbol{j}^{'}_b$. The 3×3 matrix $\boldsymbol{T}^{-1}_{\Theta}$ is then defined as $\boldsymbol{T}^{-1}_{\Theta} = [\boldsymbol{i}_b,\, \boldsymbol{j}^{'}_b,\, \boldsymbol{k}^{''}_b]$. Expressions for the unit normal vectors along the axis of the intermediate reference frames can be found by using the principal rotation matrices for the sequence of rotations described above. Consider first a coordinate $\boldsymbol{c}$ describing the location of point P in a local reference frame. Assume that this point now is observed from a reference frame rotated an angle ϕ about the local reference frame, denoted $\boldsymbol{c}^x$. These coordinates can now be related as

$$\boldsymbol{c}^x = \boldsymbol{R}_x(\phi)\boldsymbol{c} \tag{5.2}$$

where $\boldsymbol{R}_x$ is a principal rotation matrix about the local x-axis. Because the rotation matrix is orthogonal, we can write the inverse of the matrix as $\boldsymbol{R}_x^{-1} = \boldsymbol{R}_x^{\top}$ [139]. Using this, an expression for the unit normal vector along the y-axis of the first intermediate reference frame, $\boldsymbol{j}_b^{'}$ from (5.1), can be expressed in terms of the body fixed reference frame as

$$\boldsymbol{j}_b^{'} = \boldsymbol{R}_x^{\top}(\phi)\boldsymbol{j}_b \tag{5.3}$$

where $\boldsymbol{j}_b = [0,\ 1,\ 0]^T$ is the unit normal vector along the y-axis of the body fixed reference frame. Similarly, the coordinate $\boldsymbol{c}^x$, can be observed from a new reference frame, rotated an angle θ about the previous reference frame, given in (5.2), denoted $\boldsymbol{c}^y$. The relation between $\boldsymbol{c}^x$ and $\boldsymbol{c}^y$ is then expressed as

$$\boldsymbol{c}^y = \boldsymbol{R}_y(\theta)\boldsymbol{c}^x \tag{5.4}$$

Using this expression, we find that the unit normal vector $\boldsymbol{k}_b^{''}$ can be expressed as

$$\boldsymbol{k}_b^{''} = \boldsymbol{R}_y^{\top}(\theta)\boldsymbol{k}_b^{'} = \boldsymbol{R}_x^{\top}\boldsymbol{R}_y^{\top}\boldsymbol{k}_b \tag{5.5}$$

where $\boldsymbol{k}_b = [0,\ 0,\ 1]^{\top}$. With these transformations defined, we can express the transformation matrix of (5.1) as

$$\boldsymbol{T}_{\Theta}^{-1}(\boldsymbol{\Theta}) = \left[\begin{array}{ccc} \boldsymbol{i}_b, & \boldsymbol{R}_x^{\top}\boldsymbol{j}_b, & \boldsymbol{R}_x^{\top}\boldsymbol{R}_y^{\top}\boldsymbol{k}_b \end{array}\right] \tag{5.6}$$

The final principal rotation matrix $\boldsymbol{R}_z(\psi)$ can be used in order to transform a coordinate expressed in terms of the second intermediate reference frame, to be expressed in terms of the inertial reference frame. We can now design the rotation matrix transforming a coordinate representation from the vehicle body fixed reference frame to the inertial reference frame as

$$\boldsymbol{R}_b^0 = \boldsymbol{R}_z(\psi)\boldsymbol{R}_y(\theta)\boldsymbol{R}_x(\phi) \tag{5.7}$$

with

$$\begin{aligned} \boldsymbol{R}_z(\psi) &= \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix},\ \boldsymbol{R}_y(\theta) = \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \\ \boldsymbol{R}_x(\phi) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix} \end{aligned} \tag{5.8}$$

where $s_x = \sin(x)$, and $c_x = \cos(x)$. We can now write

$$\boldsymbol{c}^0 = \boldsymbol{R}_b^0 \boldsymbol{c}^b \tag{5.9}$$

This rotation matrix, as with the principal rotation matrices, is orthogonal [139], such that

$$\left(\boldsymbol{R}_b^0\right)^{-1} = \left(\boldsymbol{R}_b^0\right)^\top = \boldsymbol{R}_0^b \tag{5.10}$$

With the kinematic relations in place, we can derive expressions for the kinetic energy of the vehicle.

5.2.2 Kinetic Energy of the Vehicle

The kinetic energy of the vehicle can be expressed as

$$T = \frac{1}{2}\left((\boldsymbol{v}_{cg/0}^b)^\top \boldsymbol{M} \boldsymbol{v}_{cg/0}^b + (\boldsymbol{\omega}_{b/0}^b)^\top \boldsymbol{I}_g \boldsymbol{\omega}_{b/0}^b\right) \tag{5.11}$$

where $\boldsymbol{M} = m\boldsymbol{I}_{3\times3}$, m is the mass of the vehicle, $\boldsymbol{I}_{3\times3}$ is the identity matrix, $\boldsymbol{I}_g$ is the vehicle inertia tensor, and $\boldsymbol{v}_{cg/0}$ is the linear velocity of the vehicle center of gravity relative to the inertial reference frame. However, using the Lagrangian approach, the kinetic energy should be expressed in terms of a set of generalized coordinates and their rates. The generalized coordinates are a set of coordinates that uniquely define the position and orientation of the vehicle, and are in this chapter chosen as

$$\boldsymbol{q} = \left[\left(\boldsymbol{r}_{b/0}^0\right)^T, \boldsymbol{\Theta}^\top\right]^\top \tag{5.12}$$

The linear velocity of the vehicle center of gravity can be expressed in terms of the generalized coordinates and corresponding rates as

$$\begin{aligned}\boldsymbol{v}_{cg/0}^b &= \boldsymbol{v}_{b/0}^b + \boldsymbol{\omega}_{b/0}^b \times \boldsymbol{r}_{cg/b}^b \\ &= \boldsymbol{R}_0^b(\boldsymbol{\Theta})\dot{\boldsymbol{r}}_{b/0}^0 + \boldsymbol{T}_\Theta^{-1}(\boldsymbol{\Theta})\dot{\boldsymbol{\Theta}} \times \boldsymbol{r}_{cg/b}^b\end{aligned} \tag{5.13}$$

where $\boldsymbol{v}_{b/0}^b$ is the velocity of the origin of the vehicle body fixed reference frame, and $\boldsymbol{r}_{cg/b}^b$ is the vector from the origin of the vehicle body fixed reference frame to the center of gravity. By substituting (5.1) and (5.13) in (5.11), the kinetic energy takes the form $T(\boldsymbol{q}, \dot{\boldsymbol{q}})$. We do however seek to replace the dependency on $\dot{\boldsymbol{q}}$ by the quasi coordinates given as

$$\boldsymbol{\omega} = \begin{bmatrix} \boldsymbol{v}_{b/0}^b \\ \boldsymbol{\omega}_{b/0}^b \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_0^b & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{T}_\Theta^{-1} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{r}}_{b/0}^0 \\ \dot{\boldsymbol{\Theta}} \end{bmatrix} = \boldsymbol{\alpha}^\top \dot{\boldsymbol{q}} \tag{5.14}$$

because this will make the resulting equations of motion dependent on the linear and angular velocity in terms of the body frame, rather than the linear velocity in terms of the inertial frame and the Euler angle rates. The inverse of (5.14) is

$$\dot{\boldsymbol{q}} = \boldsymbol{\beta}\boldsymbol{\omega} \tag{5.15}$$

where

$$\boldsymbol{\beta} = (\boldsymbol{\alpha}^\top)^{-1} = \begin{bmatrix} \boldsymbol{R}_b^0 & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{T}_\Theta \end{bmatrix} \tag{5.16}$$

Substituting (5.15) into the expression for $T(\boldsymbol{q}, \dot{\boldsymbol{q}})$, yields the expression $T(\boldsymbol{q}, \boldsymbol{\beta}\boldsymbol{\omega}) = \bar{T}(\boldsymbol{q}, \boldsymbol{\omega})$, which can be found explicitly by finding the linear velocity of the vehicle center of gravity expressed in terms of the quasi-coordinates. This is recognized as the first expression in (5.13), and can be expressed compactly as

$$\begin{aligned} \boldsymbol{v}_{cg/0}^b &= \begin{bmatrix} \boldsymbol{I}_{3\times3}, & \boldsymbol{i}_b \times \boldsymbol{r}_{cg/0}^b, & \boldsymbol{j}_b \times \boldsymbol{r}_{cg/0}^b, & \boldsymbol{k}_b \times \boldsymbol{r}_{cg/0}^b \end{bmatrix} \boldsymbol{\omega} \\ &\triangleq \boldsymbol{J}_b^v \boldsymbol{\omega} \end{aligned} \tag{5.17}$$

where $\boldsymbol{J}_b^v$ is the geometric Jacobian matrix for the linear velocity of the center of gravity of the vehicle. More trivially, the angular velocity can be expressed in matrix form as

$$\begin{aligned} \boldsymbol{\omega}_{b/0}^b &= \begin{bmatrix} \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \end{bmatrix} \boldsymbol{\omega} \\ &\triangleq \boldsymbol{J}_b^\omega \boldsymbol{\omega} \end{aligned} \tag{5.18}$$

The vector $\boldsymbol{v}_b = [(\boldsymbol{v}_{cg/0}^b)^\top, (\boldsymbol{\omega}_{cg/0}^b)^\top]^\top$ collects the linear velocity of the center of gravity of the vehicle and the angular velocity of the body. This can be expressed as

$$\boldsymbol{v}_b = \begin{bmatrix} \boldsymbol{J}_b^v \\ \boldsymbol{J}_b^\omega \end{bmatrix} \boldsymbol{\omega} = \boldsymbol{J}_b \boldsymbol{\omega} \tag{5.19}$$

With this, we find the kinetic energy in terms of quasi coordinates as

$$\begin{aligned} \bar{T}_b(\boldsymbol{q}, \boldsymbol{\omega}) &= \frac{1}{2} \boldsymbol{\omega}^\top \boldsymbol{J}_b^\top \begin{bmatrix} \boldsymbol{M} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_g \end{bmatrix} \boldsymbol{J}_b \boldsymbol{\omega} \\ &\triangleq \frac{1}{2} \boldsymbol{\omega}^\top \boldsymbol{B}_b \boldsymbol{\omega} \end{aligned} \tag{5.20}$$

where $\boldsymbol{B}_b$ is the symmetric and positive definite vehicle mass-inertia matrix. The equations of motion are found by inserting the kinetic energy expression into the Lagrange's method.

5.2.3 Equations of Motion

In the traditional Lagrange method, in which the kinetic energy is expressed in terms of generalized coordinates and rates, as opposed to generalized coordinates and quasi-coordinates, the equations of motion takes the form

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial T}{\partial \boldsymbol{q}} = \boldsymbol{\tau} \tag{5.21}$$

where $\boldsymbol{\tau}$ is the vector of generalized forces. Note that the potential energy of the system is not included here. Examples on how these effects can be included will be given in section 5.5. When introducing quasi-coordinates, the chain rule must be used when differentiating because the quasi-coordinates are functions of the generalized coordinates and rates. From [24], we have that the quasi-equations of motion becomes

$$\frac{d}{dt}\left(\frac{\partial \bar{T}}{\partial \boldsymbol{\omega}}\right) + \boldsymbol{\beta}^\top \boldsymbol{\gamma} \frac{\partial \bar{T}}{\partial \boldsymbol{\omega}} - \boldsymbol{\beta}^\top \frac{\partial \bar{T}}{\partial \boldsymbol{q}} = \boldsymbol{\beta}^\top \boldsymbol{\tau} \tag{5.22}$$

where the $n \times n$ matrix $\boldsymbol{\gamma}$ of (5.22) is given as

$$\boldsymbol{\gamma} = \begin{bmatrix} \xi_{11} & \cdots & \xi_{1n} \\ \vdots & \ddots & \vdots \\ \xi_{n1} & \cdots & \xi_{nn} \end{bmatrix} - \begin{bmatrix} \boldsymbol{\omega}^\top \boldsymbol{\beta}^\top \frac{\partial \boldsymbol{\alpha}}{\partial q_1} \\ \vdots \\ \boldsymbol{\omega}^\top \boldsymbol{\beta}^\top \frac{\partial \boldsymbol{\alpha}}{\partial q_n} \end{bmatrix} \tag{5.23}$$

and

$$\xi_{ij} = \boldsymbol{\omega}^\top \boldsymbol{\beta}^\top \frac{\partial \alpha_{ij}}{\partial \boldsymbol{q}} \tag{5.24}$$

Note that $\partial \boldsymbol{\alpha}/\partial q_i$ is a square matrix, in which each element α_{ij} are differentiated with respect to q_i, whereas $\partial \alpha_{ij}/\partial \boldsymbol{q}$ is a column vector in which the element α_{ij} is differentiated with respect to each of the generalized coordinates.

The kinetic energy differentiated with respect to the velocity constitutes the momentum of the system in question. Thus

$$\dot{\boldsymbol{p}} = \frac{d}{dt}\left(\frac{\partial \bar{T}}{\partial \boldsymbol{\omega}}\right) \tag{5.25}$$

where $\boldsymbol{p}$ is the momentum of the quasi states, i.e., the momentum expressed in terms of the vehicle body fixed reference frame. Going back to (5.20), we find that

$$\frac{\partial \bar{T}}{\partial \boldsymbol{\omega}} = \boldsymbol{B}_b \boldsymbol{\omega} \tag{5.26}$$

Inverting (5.26) and substituting $\boldsymbol{p} = \partial \bar{T}/\partial \boldsymbol{\omega}$, yields

$$\boldsymbol{\omega} = \boldsymbol{B}_b^{-1}\boldsymbol{p} \tag{5.27}$$

We also find, by comparing (5.22) and (5.25), that

$$\begin{aligned}\dot{\boldsymbol{p}} &= -\boldsymbol{\beta}^\top\boldsymbol{\gamma}\frac{\partial \bar{T}}{\partial \boldsymbol{\omega}} + \boldsymbol{\beta}^\top\frac{\partial \bar{T}}{\partial \boldsymbol{q}} + \boldsymbol{\beta}^\top\boldsymbol{\tau} \\ &= -\boldsymbol{\beta}^\top\boldsymbol{\gamma}\boldsymbol{B}_b\boldsymbol{\omega} + \frac{1}{2}\boldsymbol{\beta}^\top\boldsymbol{\omega}^\top\frac{\partial \boldsymbol{B}_b}{\partial \boldsymbol{q}}\boldsymbol{\omega} + \boldsymbol{\beta}^\top\boldsymbol{\tau} \\ &\triangleq \boldsymbol{f}_p(\boldsymbol{q},\boldsymbol{\omega}) + \boldsymbol{\beta}^\top\boldsymbol{\tau}\end{aligned} \tag{5.28}$$

where we have used that

$$\frac{\partial \bar{T}}{\partial \boldsymbol{q}} = \frac{1}{2}\boldsymbol{\omega}^\top\frac{\partial \boldsymbol{B}_b}{\partial \boldsymbol{q}}\boldsymbol{\omega} \tag{5.29}$$

and

$$\boldsymbol{\omega}^\top\frac{\partial \boldsymbol{B}_b}{\partial \boldsymbol{q}}\boldsymbol{\omega} = \begin{bmatrix} \boldsymbol{\omega}^\top\frac{\partial \boldsymbol{B}_b}{\partial q_1} \\ \vdots \\ \boldsymbol{\omega}^\top\frac{\partial \boldsymbol{B}_b}{\partial q_n}\end{bmatrix}\boldsymbol{\omega} \tag{5.30}$$

Note that, in the case of the marine vehicle, the system mass-inertia matrix is not a function of the generalized coordinates, so $\partial T/\partial \boldsymbol{q} = 0$. We have however included the expression because this, in general, will not be the case when e.g. a crane is added to the system.

Combining (5.27) and (5.28) we obtain a state space model describing the basic dynamics of the vehicle, as

$$\boldsymbol{\omega} = \boldsymbol{B}_b^{-1}\boldsymbol{p} \tag{5.31a}$$

$$\dot{\boldsymbol{q}} = \boldsymbol{\beta}\boldsymbol{\omega} \tag{5.31b}$$

$$\dot{\boldsymbol{p}} = \boldsymbol{f}_p(\boldsymbol{q},\boldsymbol{\omega}) + \boldsymbol{\beta}^\top\boldsymbol{\tau} \tag{5.31c}$$

We have now derived state space equations for the vehicle. In the following section this formulation is expanded to also include crane of manipulator dynamics. With the state space equations for the vehicle derived, we can, in the following section expand these to include the crane or manipulator dynamics.

5.3 Expanding the Model to Include Crane and Manipulator Dynamics

The system, now defined as the vehicle and the cranes or manipulators, move in $n = 6 + k$ degrees of freedom, where the vehicle move in 6 degrees of freedom, and the equipment in k. In case of lower pair jointed, open chain structured equipment, this means that the equipment have k joints. For such equipment, the obvious choice for generalized coordinates are the joint displacements, denoted $\boldsymbol{q}_e = [q_{e1}, q_{e2}, ..., q_{ek}]^\top$. The system vector of generalized coordinates are thus the $n \times 1$ vector $\boldsymbol{q} = [(\boldsymbol{r}^0_{b/0})^\top, \boldsymbol{\Theta}^\top, \boldsymbol{q}_e^T]^\top$. The quasi coordinates of the equipment are defined simply as the rate of the generalized coordinates of the equipment, such that the system vector of quasi-coordinates are $\boldsymbol{\omega} = [(\boldsymbol{v}^b_{b/0})^\top, (\boldsymbol{\omega}^b_{b/0})^\top, \dot{\boldsymbol{q}}_e^\top]^\top$. With these augmented vectors of generalized coordinates and quasi-coordinates, it is necessary to augment the transformation matrices $\boldsymbol{\alpha}^\top$, and $\boldsymbol{\beta}$. Recall that we had $\boldsymbol{\omega} = \boldsymbol{\alpha}^\top \dot{\boldsymbol{q}}$. Using the expression (5.14), together with the notation $\dot{\boldsymbol{q}}_e = \boldsymbol{I}_{k\times k}\dot{\boldsymbol{q}}_e$, we find that the augmented $n \times n$ transformation matrix $\boldsymbol{\alpha}^\top$ is given as

$$\boldsymbol{\alpha}^\top(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R}^b_0 & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times k} \\ \mathbf{0}_{3\times 3} & \boldsymbol{T}^{-1}_\Theta & \mathbf{0}_{3\times k} \\ \mathbf{0}_{k\times 3} & \mathbf{0}_{k\times 3} & \boldsymbol{I}_{k\times k} \end{bmatrix} \tag{5.32}$$

The augment inverse transformation matrix is then

$$\boldsymbol{\beta}(\boldsymbol{q}) = \left(\boldsymbol{\alpha}^\top\right)^{-1} = \begin{bmatrix} \boldsymbol{R}^0_b & \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times k} \\ \mathbf{0}_{3\times 3} & \boldsymbol{T}_\Theta & \mathbf{0}_{3\times k} \\ \mathbf{0}_{k\times 3} & \mathbf{0}_{k\times 3} & \boldsymbol{I}_{k\times k} \end{bmatrix} \tag{5.33}$$

Before deriving the equations of motion, we shall investigate the kinematics of the system. In particular, we seek to find expressions for the velocity of the centre of gravity for each of the equipment bodies as functions of the generalized coordinates and the quasi-coordinates, in order to find an expression for the system kinetic energy. To this purpose it is necessary to find expressions for the coordinates of each of the bodies' centre of gravity, relative to the preceding joints and the body fixed reference frame.

Figure 5.2 shows some equipment with an open chain structure, e.g. a robotic manipulator. In this case, there are two revolute joints and one prismatic joint. In each joint, there is a reference frame attached to the corresponding body such that body i is attached to reference frame i. If joint i is a revolute joint, body i rotate about the vector $\boldsymbol{e}_i$, and if joint i is a prismatic joint, body i displace along the vector $\boldsymbol{e}_i$. For the sake of convenience, we place the reference frames such that the rotation or displacement of joint i takes place about or along one of the principal axis of the local reference frame. In the following, we assume that the location of the centre of gravity of each link relative to the link reference frame origin is known. For link i these coordinates are denoted $\boldsymbol{r}^i_{cgi/i}$. We also define the coordinates of joint $i+1$ relative to joint i, in terms of reference frame i, as $\boldsymbol{r}^i_{i+1/i}$. In the case when joint i is a revolute joint, these coordinates are constant, and in the case of prismatic joints, the coordinates are dependent on the displacement $q_{e(i+1)}$.

In order to find the coordinates $\boldsymbol{r}^i_{i+1/i}$ in this case, we define the coordinate $\boldsymbol{r}^i_{zi/i}$, as the point where reference frame $i + 1$ is located for $q_{e(i+1)} = 0$, relative to reference frame i. An expression for the vector $\boldsymbol{r}_{i+1/i}$ in the case of joint $i + 1$ being prismatic is then

$$\boldsymbol{r}_{i+1/i} = \boldsymbol{r}_{zi/i} + \boldsymbol{e}_{i+1} q_{i+1} \tag{5.34}$$

With the coordinates $\boldsymbol{r}_{cgi/i}$, $\boldsymbol{r}_{i+1/i}$, and the coordinate of the first link relative to the origin of the body fixed reference frame $\boldsymbol{r}_{1/b}$, we can find the coordinates of any center of gravity, relative to any joint, as well as relative to the body fixed reference frame of the vehicle. As an example, the position of the center of gravity for body i, relative to the origin of the body fixed reference frame is

$$\boldsymbol{r}_{cgi/b} = \boldsymbol{r}_{1/b} + \boldsymbol{r}_{2/1} + ... + \boldsymbol{r}_{i/i-1} + \boldsymbol{r}_{cgi/i} \tag{5.35}$$

We do however need to express all the terms in (5.35) in terms of the same reference frame. To this purpose we develop rotation matrices as functions of the generalized coordinates, mapping vectors expressed in terms of any of the local reference frames into a reference frame with the same orientation as the vehicle body fixed reference frame. The notation used for the rotation matrices is illustrated in 5.36, where a vector $\boldsymbol{c}$ expressed in terms of reference frame i is transformed to a representation in the reference frame b as

$$\boldsymbol{c}^b = \boldsymbol{R}^b_i(\boldsymbol{q}_e)\boldsymbol{c}^i \tag{5.36}$$

Next, we investigate the differential kinematics of the system in order to develop geometric Jacobian matrices as functions of the generalized coordinates. These matrices, one for each body of the equipment, maps the quasi-coordinates into the angular and linear velocity of the centre of gravity of the given body, equivalent to (5.19), where the vehicle geometric Jacobian is defined.

5.3.1 Differential Kinematics

Both the linear and angular velocities of the various bodies of a chain of linked rigid bodies situated on a marine vehicle, are explicitly dependent on the velocity of the vehicle and the rates of the preceding joints. We define the contribution to the linear velocity of the centre of gravity of body i from the linear velocity of the vehicle as

$$\begin{aligned} \boldsymbol{v}^{(\boldsymbol{v}_{b/0})}_{cgi/0} &= \boldsymbol{v}^b_{b/0} = \boldsymbol{I}_{3\times3}\boldsymbol{v}^b_{b/0} \\ &\triangleq \boldsymbol{\mathcal{J}}^{\boldsymbol{v}_{cgi}}_{\boldsymbol{v}_b}\boldsymbol{v}^b_{b/0} \end{aligned} \tag{5.37}$$

where the superscript in parenthesis denotes from where the given contribution comes.

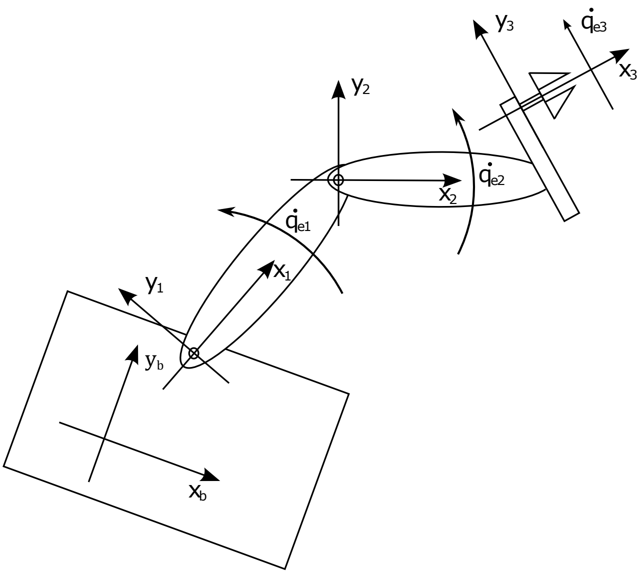


Figure 5.2: Kinematics of an open chain of linked bodies.

The contribution to the same velocity, by the angular velocity of the vehicle is defined as

$$\begin{aligned}\boldsymbol{v}_{cgi/0}^{(\omega_{b/0})} &= \boldsymbol{\omega}_{b/0}^{b} \times \boldsymbol{r}_{cgi/b}^{b} \\ &= \left(\boldsymbol{\omega}_{b/0}^{b}\boldsymbol{i}_b + \boldsymbol{\omega}_{b/0}^{b}\boldsymbol{j}_b + \boldsymbol{\omega}_{b/0}^{b}\boldsymbol{k}_b\right) \times \boldsymbol{r}_{cgi/b}^{b} \\ &= \left[\boldsymbol{i}_b \times \boldsymbol{r}_{cgi/b}^{b},\ \boldsymbol{j}_b \times \boldsymbol{r}_{cgi/b}^{b},\ \boldsymbol{k}_b \times \boldsymbol{r}_{cgi/b}^{b}\right]\boldsymbol{\omega}_{b/0}^{b} \\ &\triangleq \boldsymbol{\mathcal{J}}_{\omega_b}^{\boldsymbol{v}_{cgi}}\boldsymbol{\omega}_{b/0}^{b}\end{aligned} \tag{5.38}$$

where $\boldsymbol{r}_{cgi/b}$ is the coordinate of the center of gravity of body i relative to the origin of the vehicle body fixed reference frame. The contribution to the linear velocity of the i-th center of gravity from the rate of joint p for $p \leqslant i$, depends on whether the joint is revolute or prismatic. We define

$$\begin{aligned}\boldsymbol{v}_{cgi/0}^{(\dot{q}_{ep})} &\triangleq \boldsymbol{\mathcal{J}}_{\dot{q}_{ep}}^{\boldsymbol{v}_{cgi}}\dot{q}_{ep} \\ &= \begin{cases}(\boldsymbol{e}_p^b \times \boldsymbol{r}_{cgi/p}^{b})\dot{q}_{ep}, & \text{for revolute} \\ \boldsymbol{e}_p^b\dot{q}_{ep} & \text{for prismatic}\end{cases}\end{aligned} \tag{5.39}$$

where $\boldsymbol{r}_{cgi/p}$ is the coordinate of the center of gravity of body i relative to the origin of reference frame p, and $\boldsymbol{e}_p^b$ is the vector about, or along, which body p revolves or translates, respectively, in terms of the vehicle body fixed reference frame. In the case where $p > i$, the contribution is zero. Using (5.37) through (5.39), we can find the linear velocity of the center of gravity of link i, expressed in terms of the vehicle body fixed reference frame, as a function of the generalized coordinates and the quasi-coordinates, as

$$\begin{aligned} \boldsymbol{v}^b_{cgi/0} &= \left[\boldsymbol{\mathcal{J}}^{\boldsymbol{v}_{cgi}}_{\boldsymbol{v}_b},\, \boldsymbol{\mathcal{J}}^{\boldsymbol{v}_{cgi}}_{\boldsymbol{\omega}_b},\, \boldsymbol{\mathcal{J}}^{\boldsymbol{v}_{cgi}}_{\dot{q}_{e1}},\, \ldots,\, \boldsymbol{\mathcal{J}}^{\boldsymbol{v}_{cgi}}_{\dot{q}_{ei}},\, \mathbf{0}\right]\boldsymbol{\omega} \\ &\triangleq \boldsymbol{J}^v_i(\boldsymbol{q})\boldsymbol{\omega} \end{aligned} \tag{5.40}$$

where the dimensions of the zero matrix **0** is $3 \times (k - i)$.

We now proceed to find the various contributions to the angular velocity of body i. There is no contribution to this velocity from the linear velocity of the vehicle. Thus, we can define

$$\begin{aligned} \boldsymbol{\omega}^{(\boldsymbol{v}_b)}_i &= \mathbf{0}_{3\times 3}\boldsymbol{v}^b_{b/0} \\ &\triangleq \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\boldsymbol{v}_b}\boldsymbol{v}^b_{b/0} \end{aligned} \tag{5.41}$$

The contribution from the angular velocity of the vehicle can be formulated as

$$\begin{aligned} \boldsymbol{\omega}^{(\boldsymbol{\omega}_{b/0})}_i &= \boldsymbol{I}_{3\times 3}\boldsymbol{\omega}^b_{b/0} \\ &\triangleq \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\boldsymbol{\omega}_b}\boldsymbol{\omega}^b_{b/0} \end{aligned} \tag{5.42}$$

Finally, the contribution to the angular velocity from the joint displacement rate $\dot{q}_{ep}$, given that $p \leqslant i$, is

$$\begin{aligned} \boldsymbol{\omega}^{(\dot{q}_{ep})}_i &\triangleq \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\dot{q}_{ep}}\dot{q}_{ep} \\ \Rightarrow \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\dot{q}_{ep}} &= \begin{cases} \boldsymbol{e}^b_p, & \text{for revolute} \\ \mathbf{0}_{3\times 1} & \text{for prismatic} \end{cases} \end{aligned} \tag{5.43}$$

The total angular velocity of body i of the equipment, can be found by taking the sum of all contributions stated in (5.41) through (5.43) as

$$\begin{aligned} \boldsymbol{\omega}^b_{i/0} &= \left[\boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\boldsymbol{v}_b},\, \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\boldsymbol{\omega}_b},\, \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\dot{q}_{e1}},\, \ldots,\, \boldsymbol{\mathcal{J}}^{\boldsymbol{\omega}_i}_{\dot{q}_{ei}},\, \mathbf{0}\right]\boldsymbol{\omega} \\ &\triangleq \boldsymbol{J}^\omega_i(\boldsymbol{q})\boldsymbol{\omega} \end{aligned} \tag{5.44}$$

where the zero matrix is of dimension $3 \times (k - i)$. We define the 6×1 vector $\boldsymbol{v}_i = [(\boldsymbol{v}^b_{cgi/0})^\top, (\boldsymbol{\omega}^b_{i/0})^\top]^\top$, where the linear and angular velocity of the center of gravity of body i are collected. Furthermore, we define the $6 \times n$ geometric Jacobian matrix for the velocity of body i as

$$\boldsymbol{J}_i(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{J}_i^v(\boldsymbol{q}) \\ \boldsymbol{J}_i^\omega(\boldsymbol{q}) \end{bmatrix} \tag{5.45}$$

Using this, a compact expression for the velocity of the center of gravity for body i is

$$\boldsymbol{v}_i = \boldsymbol{J}_i(\boldsymbol{q})\boldsymbol{\omega} \tag{5.46}$$

These kinetic relations can further be used in order to derive kinetic energy expressions for the system.

5.3.2 Kinetic Energy of System

The kinetic energy the system can be found by taking the sum of all contributions from each body in the system. In (5.20), the contribution to the total kinetic energy from the vehicle is found. It is however necessary to augment this expression, as $\boldsymbol{q}$ and $\boldsymbol{\omega}$ have been augmented. This is achieved by augmenting the geometric Jacobian matrix found in (5.19) to

$$\boldsymbol{J}_b = \begin{bmatrix} \boldsymbol{J}_b^v & \boldsymbol{0}_{3\times k} \\ \boldsymbol{J}_b^\omega & \boldsymbol{0}_{3\times k} \end{bmatrix} \tag{5.47}$$

in order to make it compatible to the new vector of quasi-coordinates.

The kinetic energy of body i of the crane or manipulator can, as for the vehicle, be found as

$$\begin{aligned} \bar{T}_i(\boldsymbol{q},\boldsymbol{\omega}) &= \frac{1}{2}\boldsymbol{\omega}^\top \boldsymbol{J}_i^\top(\boldsymbol{q}) \begin{bmatrix} \boldsymbol{M}_i & \boldsymbol{0}_{3\times 3} \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{I}_i^b \end{bmatrix} \boldsymbol{J}_i(\boldsymbol{q})\boldsymbol{\omega} \\ &\triangleq \frac{1}{2}\boldsymbol{\omega}^\top \boldsymbol{B}_i(\boldsymbol{q})\boldsymbol{\omega} \end{aligned} \tag{5.48}$$

where $\boldsymbol{M}_i = m_i \boldsymbol{I}_{3\times 3}$, m_i is the mass of body i, and $\boldsymbol{I}_i^b = \boldsymbol{R}_i^b \boldsymbol{I}_i \boldsymbol{R}_b^i$ is the inertia tensor of body i, expressed in terms of the vehicle body fixed reference frame. The matrix $\boldsymbol{I}_i$ is the locally expressed inertia tensor, and $\boldsymbol{B}_i(\boldsymbol{q})$ is the equipment body i mass-inertia matrix, which also is symmetric and positive definite.

To find the system kinetic energy, we take the sum of all contributions as

$$\begin{aligned}\bar{T}(\boldsymbol{q},\boldsymbol{\omega}) &= \bar{T}_b(\boldsymbol{q},\boldsymbol{\omega}) + \sum_{i=1}^{k}\left(\bar{T}_i(\boldsymbol{q},\boldsymbol{\omega})\right)\\ &= \frac{1}{2}\boldsymbol{\omega}^\top\left(\boldsymbol{B}_b + \sum_{i=1}^{k}\left(\boldsymbol{B}_i(\boldsymbol{q})\right)\right)\boldsymbol{\omega}\\ &\triangleq \frac{1}{2}\boldsymbol{\omega}^\top\boldsymbol{B}(\boldsymbol{q})\boldsymbol{\omega}\end{aligned} \tag{5.49}$$

where the symmetric and positive definite system mass-inertia matrix $\boldsymbol{B}(\boldsymbol{q})$ is the sum of the individual bodies mass-inertia matrices.

Using the equations (5.27) and (5.28), we find a state space model for the complete system as

$$\boldsymbol{\omega} = \boldsymbol{B}^{-1}\boldsymbol{p} \tag{5.50a}$$

$$\dot{\boldsymbol{q}} = \boldsymbol{\beta}\boldsymbol{\omega} \tag{5.50b}$$

$$\dot{\boldsymbol{p}} = \boldsymbol{f}_p(\boldsymbol{q},\boldsymbol{\omega}) + \boldsymbol{\beta}^\top\boldsymbol{\tau} \tag{5.50c}$$

The derived state space model describing the system can now be implemented in the bond graph framework.

5.4 Bond Graph Implementation

We now have a set of equations describing the basic dynamics of the system, i.e., the dynamics of the system related to the kinetic energy of the system of bodies. This set of equations is well suited for implementation in the bond graph language. After creating a bond graph template of the system, i.e, implementing (5.50), interfaces between the template and subsystems are discussed in a general manner, before we, in the next section, introduce a case study, where examples of such subsystems and interfaces are demonstrated. At this point, gravity forces and restoring forces, along with other subsystems, are included.

5.4.1 Basic Model

The equations in (5.50) can be implemented in a bond graph as shown in figure 5.3. The set of equations is dependent on the generalized coordinates $\boldsymbol{q}$, the quasi coordinates $\boldsymbol{\omega}$, and the momentum $\boldsymbol{p}$. The implementation to the left in Figure 5.3, shows three vector power bonds sharing the same 1-junction. By letting the effort $\boldsymbol{e}_1 = \dot{\boldsymbol{p}}_1$, and the flow $\boldsymbol{f}_2 = \boldsymbol{\omega}_2$, be input ports to the IC-field, we seek to find expressions for the outputs $\dot{\boldsymbol{p}}_2$ and $\boldsymbol{\omega}_1$. Note that the subscript notation in this figure does not indicate certain elements of the vector, but the numbers assigned to

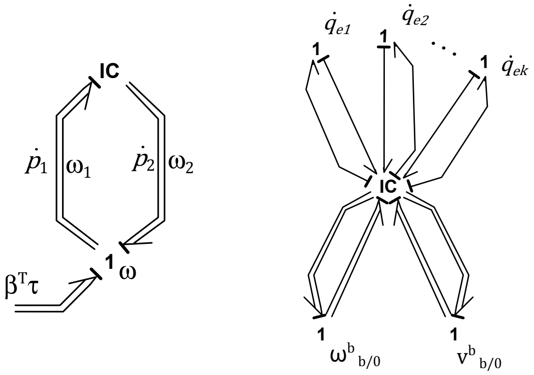


Figure 5.3: Left figure: Basic bond graph of the system with single 1-junction. Right figure: Basic bond graph with flows separated into several 1-junctions

the power bonds. As all three power bonds are connected to the same 1-junction, we have that $\boldsymbol{\omega}_1 = \boldsymbol{\omega}_2 = \boldsymbol{\omega}$ and $\dot{\boldsymbol{p}}_1 = \dot{\boldsymbol{p}}_2 + \boldsymbol{\beta}^\top \boldsymbol{\tau} = \dot{\boldsymbol{p}}$. Thus, the constitutive relations for the IC-field are

$$\begin{aligned} \boldsymbol{\omega}_1 &= \boldsymbol{B}^{-1}(\boldsymbol{q})\boldsymbol{p}_1 \\ \dot{\boldsymbol{p}}_2 &= \boldsymbol{f}_p(\boldsymbol{q}, \boldsymbol{\omega}_1) \end{aligned} \tag{5.51}$$

where the vector of generalized coordinates is found by integrating the equation

$$\dot{\boldsymbol{q}} = \boldsymbol{\beta}\boldsymbol{\omega} \tag{5.52}$$

In order to conveniently develop an interface extensions to this basic model, we partition the quasi-coordinate vector into the linear velocity of the vehicle, $\boldsymbol{v}^b_{b/0}$, the angular velocity of the vehicle, $\boldsymbol{\omega}^b_{b/0}$, and the joint rates of the equipment $\dot{\boldsymbol{q}}_e$. Furthermore, it might be convenient to partition the vector of joint rates into k separate velocities $\dot{q}_{e1}$, $\dot{q}_{e2}$, ..., $\dot{q}_{ek}$. We can now create separate 1-junctions, representing each of these velocity components, and connect each to the IC-field as shown to the right in Figure 5.3.

5.4.2 Connectivity

As can be seen in Figure 5.3, the basic template can be interfaced by a subsystem setting an effort expressed as a generalized force in terms of quasi-coordinates, i.e., $\boldsymbol{\beta}^\top \boldsymbol{\tau}$. The basic template then responds with a flow in terms of quasi-coordinates, i.e., $\boldsymbol{\omega}$. For most purposes however, the modeller does not need to consider this explicitly. Consider for example two subsystems, the first exerting a force $\boldsymbol{F}_p$ at the point p, and the second exerting a torque $\boldsymbol{T}_k$ at a point k on the system. The procedure for transforming this force and torque into the vectors $\boldsymbol{\beta}^\top \boldsymbol{\tau}_p$ and $\boldsymbol{\beta}^\top \boldsymbol{\tau}_k$

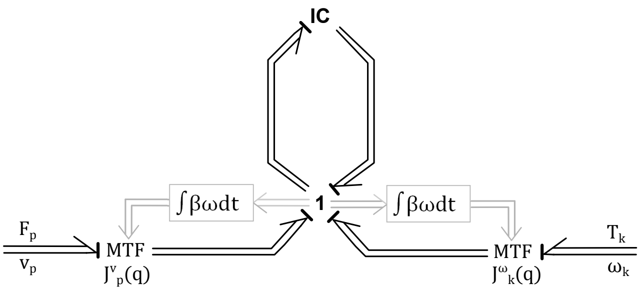


Figure 5.4: Bond graph with interface to external force and torque.

of generalized coordinates in terms of the quasi-coordinates is straight forward within the bond graph framework. This can be achieved by placing a 1-junction representing a linear velocity $\boldsymbol{\nu}_p$ for the force, and a one junction representing $\boldsymbol{\omega}_k$ for the torque, and then connecting the subsystems directly to the respective 1-junctions. The relations between the 1-junctions representing $\boldsymbol{\nu}_p$ and $\boldsymbol{\omega}_k$, and the quasi-coordinates can always be made by a modulated transformer as shown in Figure 5.4. The constitutive relations for the modulated transformers are

$$\begin{aligned} \boldsymbol{\nu}_p &= \boldsymbol{J}_p(\boldsymbol{q})\boldsymbol{\omega} \\ \boldsymbol{\beta}^\top \boldsymbol{\tau}_p &= \boldsymbol{J}_p^\top(\boldsymbol{q})\boldsymbol{F}_p \end{aligned} \tag{5.53}$$

and

$$\begin{aligned} \boldsymbol{\omega}_k &= \boldsymbol{J}_k(\boldsymbol{q})\boldsymbol{\omega} \\ \boldsymbol{\beta}^\top \boldsymbol{\tau}_k &= \boldsymbol{J}_k^\top(\boldsymbol{q})\boldsymbol{T}_k \end{aligned} \tag{5.54}$$

where the matrices $\boldsymbol{J}_p(\boldsymbol{q})$ and $\boldsymbol{J}_k(\boldsymbol{q})$ can be found in a similar manner as have been done in section 5.3.1. Figure 5.4 illustrates this concept. Gravity and buoyancy forces can be connected in this manner.

In the following, a case study, utilizing the bond graph template, along with other subsystems to demonstrate connectivity, will be presented.

5.5 Case Study - Offshore Installation Vessel with Crane

In the previous sections of this chapter a generic framework for bond graph implementation of the interconnected dynamics of marine vehicles and equipment consisting of multiple rigid bodies, such as manipulators and cranes, was derived. We have previously argued that one of the advantages of implementing this framework in bond graphs is that this facilitates well structured and well defined interfacing with models of relevant subsystems. In this section, a case-study is presented

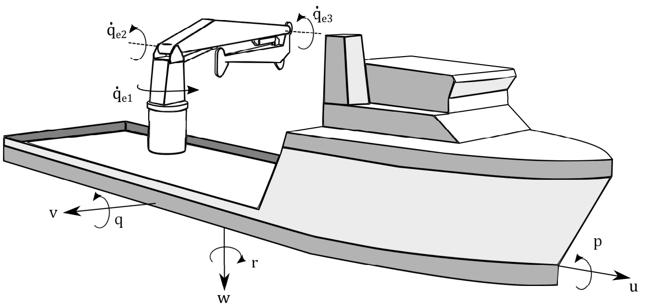


Figure 5.5: Illustration of ship and crane system.

Table 5.1: Ship and crane parameters.

Parameter	**Value**	
Length of ship	107	m
Width of ship	22	m
Height of ship	10	m
Mass of ship	2350	tons
Height of crane link 1	7	m
Mass of crane link 1	10	tons
Length of crane link 2	15	m
Mass of crane link 2	6	tons
Length of crane link 3	10	m
Mass of crane link 3	3	tons

in order to demonstrate this. In particular, a simulation model of an offshore installation vessel with a three degrees of freedom heavy-duty crane mounted on the after-deck, as shown in Figure 5.5, is presented. The intention is not to provide a state of the art marine vessel simulator, but rather to demonstrate how a vessel simulator with interconnected vessel-crane dynamics can be built based on the frame-work. In other words, we seek to demonstrate how the basic rigid-body dynamics of the vessel can be placed in an environment, i.e., how environmental forces can be connected to the model, and how the vessel and the crane can be equipped with relevant equipment such as actuators and controllers. The following subsystems are modelled and interfaced to the basic rigid body dynamics; (i) gravitational -and buoyancy forces, (ii) environmental forces, (iii) added mass and hydrodynamic damping, (iv) a thruster system for the vessel with simple thruster controllers, (v) a DP-control system providing reference signals for the thruster system, also including a nonlinear passive observer and a reference model, (vi) a wire including a payload connected to the crane, and finally, (vii) simple actuators as well as a control system for the crane. An overview of the developed bong graph model is presented in Figure 5.6.

The model is simulated using the 20-sim software [140]. We do, however, stress that the bond graph model can be simulated in any software supporting scripting since the bond graphs easily provides the state equations. One alternative, would for example be to extract equations of motion from Figure 5.6 by hand and integrate them using Python, C, Matlab, or any other software capable of simulating a set of first order ordinary differential equations. Another alternative would be to transform Figure 5.6 into a block diagram and use Matlab Simulink to simulate the system. An advantage with software that supports bond graphs, is that one avoids the tedious task of extracting the equations by hand, or transforming the bond graph into a block diagram.

For simplicity, the ship is modeled as a rectangular barge. The main dimensions for both the ship and the crane are summarized in Table 5.1.

5.5.1 Gravitational and Buoyancy Forces

Restoring forces are the forces and torques resulting from the the weight and buoyancy forces acting on the vessel. In other words, these are the forces and torques which would have been derived from the potential energy function, had it been included when deriving the Lagrangian equations. The linear restoring force, i.e., the restoring force associated to the linear motion of the vehicle, is the resulting force from the difference between the weight and the buoyancy force, while the torques appear when the centre of gravity and the centre of buoyancy are not aligned along a vertical line.

The ship is modelled as a rectangular barge, and the displaced volume is assumed to be given as $A_w z_d$, where A_w is the waterline area and z_d is the draught. Then, the buoyancy force can be expressed as

$$\boldsymbol{f}_b^0 = A_w \rho_w g z_d \tag{5.55}$$

Note that ρ_w is the density of the water and is used consistently during the whole case study. In this case, energy will be stored as a function of the vertical position of the vehicle relative to the water surface, and as such, a compliance element is the natural choice for bond graph implementation. The weight of the vessel is implemented as an effort source with the constant effort $\boldsymbol{e} = [0,\ 0,\ mg]^\top$. The linear restoring forces are expressed in terms of the inertial reference frame, while the rigid-body-system is expressed in terms of b. Therefore, a rotation transformation

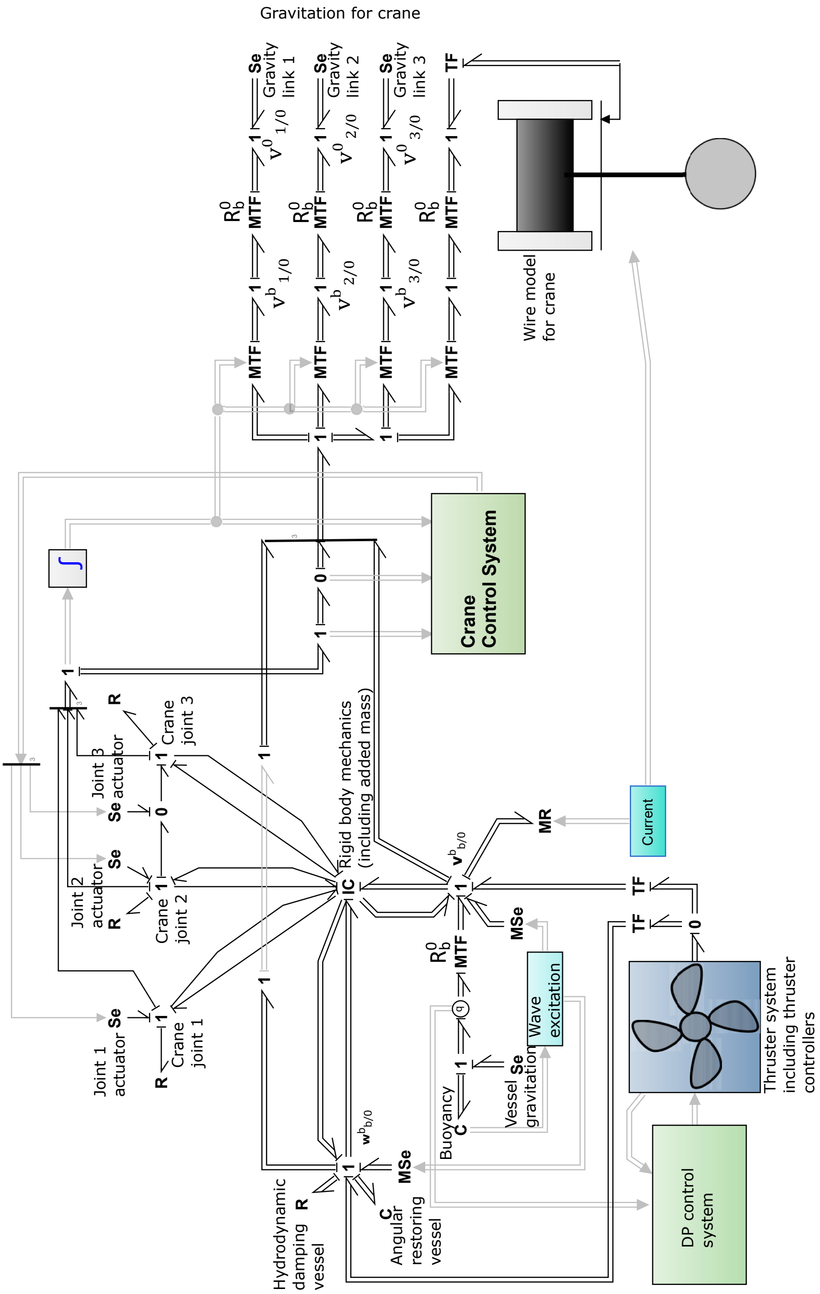


Figure 5.6: Overview of case study model.

is made between the linear restoring and the 1-junction representing $\boldsymbol{v}^b_{b/0}$, as seen in Figure 5.6.

The restoring torques acting on the vehicle body are denoted $\boldsymbol{\tau}^b_{R,b}$. These torques are expressed in terms of the vehicle body fixed reference frame and can in general be found as [14]

$$\boldsymbol{\tau}^b_{R,b} = \boldsymbol{r}^b_{cg/b} \times \boldsymbol{R}^b_0 \boldsymbol{f}^0_g + \boldsymbol{r}^b_{cb/b} \times \boldsymbol{R}^b_0 \boldsymbol{f}^0_b \tag{5.56}$$

where $\boldsymbol{r}^b_{cg/b}$ and $\boldsymbol{r}^b_{cb/b}$ are the coordinates of the the vehicle centre of gravity and centre of buoyancy relative to the origin of the vehicle body fixed reference frame, and $\boldsymbol{f}^0_g$ and $\boldsymbol{f}^0_b$ are the weight and buoyancy of the vehicle. Energy will be stored as a function of the vehicle displacement due to the restoring torques, and the compliance element is thus a suitable implementation. In Figure 5.6, the restoring torque of the vehicle body is represented by the C-element connected to the 1-junction representing the body fixed angular velocity.

In order to include weight to the crane links we place 1-junctions representing the linear velocity of the centre of gravity for each link. To the right in Figure 5.6 it is shown how the quasi-coordinates can be used in order to find these velocities using the transformation given in (5.40). The gravity forces are modelled as the rightmost effort sources. We now proceed to include added mass in the model.

5.5.2 Added Mass and Hydrodynamic Damping for the Vessel

The added mass can be shown to be a function of excitation frequency [142]. In this case study however, it is assumed to be constant and frequency independent, which according to [130, ch. 6] is a good assumption in manoeuvring theory. The added mass is included in the system by modifying the IC-field constitutive relations according to

$$\begin{aligned} \boldsymbol{\omega} &= \left[\boldsymbol{B}(\boldsymbol{q}) + \boldsymbol{B}_A\right]^{-1} \boldsymbol{p} \\ \dot{\boldsymbol{p}} &= \boldsymbol{f}_p(\boldsymbol{q}, \boldsymbol{\omega}) + \boldsymbol{C}_A(\boldsymbol{\omega})\boldsymbol{\omega} \end{aligned} \tag{5.57}$$

where $\boldsymbol{B}_A$ is the added mass matrix and $\boldsymbol{C}_A$ is the added Coriolis and centrifugal matrix. The added mass matrix, with the assumption of frequency independence, can be simplified according to [14] as

$$\begin{aligned}\boldsymbol{B}_A &= \begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} & \mathbf{0}_{1\times3} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} & \mathbf{0}_{1\times3} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} & \mathbf{0}_{1\times3} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} & \mathbf{0}_{1\times3} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} & \mathbf{0}_{1\times3} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} & \mathbf{0}_{1\times3} \\ \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \mathbf{0}_{3\times3} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}\end{aligned} \tag{5.58}$$

where the $\mathbf{0}$-vectors extensions are included in order to account for the degrees of freedom associated to the crane. The added mass effect can, according to [14, p. 119], be further simplified by assuming that the off-diagonal elements are negligible, such that

$$\begin{aligned}\boldsymbol{A}_{11} &= \begin{bmatrix} X_{\dot{u}} & 0 & 0 \\ 0 & Y_{\dot{v}} & 0 \\ 0 & 0 & Z_{\dot{w}} \end{bmatrix}, \boldsymbol{A}_{12} = \boldsymbol{A}_{21} = \mathbf{0} \\ \boldsymbol{A}_{22} &= \begin{bmatrix} K_{\dot{p}} & 0 & 0 \\ 0 & M_{\dot{q}} & 0 \\ 0 & 0 & N_{\dot{r}} \end{bmatrix}\end{aligned} \tag{5.59}$$

Here, $X_{\dot{u}}$, is the added mass in surge due to motion in the surge direction, $Y_{\dot{v}}$ is added mass in sway due to motion in the sway direction and so forth. The Coriolis and centrifugal matrix due to added mass can be found as [14]

$$\boldsymbol{C}_A(\boldsymbol{\omega}) = \begin{bmatrix} \mathbf{0} & -S(\boldsymbol{A}_{11}\boldsymbol{v}^b_{b/0} + \boldsymbol{A}_{12}\boldsymbol{\omega}^b_{b/0}) & \mathbf{0} \\ -S(\boldsymbol{A}_{11}\boldsymbol{v}^b_{b/0} + \boldsymbol{A}_{12}\boldsymbol{\omega}^b_{b/0}) & -S(\boldsymbol{A}_{21}\boldsymbol{v}^b_{b/0} + \boldsymbol{A}_{22}\boldsymbol{\omega}^b_{b/0}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{5.60}$$

where $\mathbf{0}$ are 3×3 matrices, and $S(\boldsymbol{x})$ is the cross product operator.

The hydrodynamic damping force $\boldsymbol{\tau}^b_{fv}$, and torque $\boldsymbol{\tau}^b_{f\omega}$, acting on a marine vessel in terms of the body fixed reference frame can be expressed as

$$\begin{aligned}\boldsymbol{\tau}^b_{fv} &= \boldsymbol{D}_{NLv}(\boldsymbol{v}^b_{b/c}) + \boldsymbol{D}_{Lv}\boldsymbol{v}^b_{b/c} \\ \boldsymbol{\tau}^b_{f\omega} &= \boldsymbol{D}_{NL\omega}(\boldsymbol{\omega}^b_{b/0}) + \boldsymbol{D}_{L\omega}\boldsymbol{\omega}^b_{b/0}\end{aligned} \tag{5.61}$$

where $\boldsymbol{v}_{b/c}^{b}$ is the velocity of the vessel relative to the velocity of the water particles due to current, $\boldsymbol{D}_{NLv}(\boldsymbol{v}_{b/c}^{b})$ and $\boldsymbol{D}_{NL\omega}(\boldsymbol{\omega}_{b/0}^{b})$ are non-linear damping forces, and $\boldsymbol{D}_{Lv}$ and $\boldsymbol{D}_{L\omega}$ are diagonal matrices of linear friction coefficients. Using the linear velocity of the vessel relative to the current velocity rather than the velocity relative to the inertial reference frame, automatically generates the linear forces due to current. The non-linear damping forces can comprise e.g. damping due to vortex shedding, radiation induced potential damping and wave drift damping [14]. In this case study, only non-linear vortex shedding forces and linear skin friction forces and torques are considered. The vortex shedding forces are assumed to be given as

$$\boldsymbol{D}_{NLv}(\boldsymbol{v}_{b/c}^{b}) = \frac{1}{2} C_D \rho_w \left(\boldsymbol{v}_{b/c}^{b}\right)^{\top} \boldsymbol{A}_p |\boldsymbol{v}_{b/c}^{b}| \tag{5.62}$$

where C_D is drag coefficient and $\boldsymbol{A}_p = \mathrm{diag}(A_u, A_v, A_w)$, and A_u, A_v, and A_w are the projected underwater areas in surge, sway and heave, respectively. The linear skin friction forces and torques are given as in (5.61), with diagonal coefficient matrices. The bond graph implementation of the friction forces acting on the linear velocity of the vessel can be seen as the R-element connected to $\boldsymbol{v}_{b/0}^{b}$ in Figure 5.6, and the friction forces acting on the angular velocity can be seen in the same figure as the R-element connected to the 1-junction representing $\boldsymbol{\omega}_{b/0}^{b}$.

5.5.3 Wave Excitation Forces

A ship is excited by many different environmental forces, such as forces due to the dynamic pressure field generated by waves, radiation forces, diffraction forces, and second order effects due to irregular sea. In this case study potential wave theory is used to calculate the wave induced forces and torques acting on the ship [142].

In linear wave theory the wave potential for a sine wave propagating along the x-axis is given as

$$\Phi = \frac{g\zeta_a}{\omega} e^{kz} \cos(\omega t - kx + \epsilon) \tag{5.63}$$

where ζ_a is the wave amplitude found from the Jonswap wave spectrum [142, Chapter 2, p. 25], ω is the wave frequency, g is the acceleration of gravity, k is the wave number, x is the horizontal propagation of the wave, z is the vertical distance relative to the surface with negative value below the surface and ϵ is a random phase angle. From the wave potential given in (5.63) the dynamic pressure field generated by a given wave component can be derived and expressed as

$$p_D = \rho_w \frac{\partial \Phi}{\partial t} = \rho_w g \zeta_a e^{kz} \sin(\omega t - kx + \epsilon) \tag{5.64}$$

where ρ_w is the density of the water. A realistic sea state is irregular, containing a continuum of wave components with different frequencies. In this study all wave components are calculated from the Jonswap wave spectrum and assumed to propagate from the north. By using superposition, the dynamic pressure field for an irregular sea state can be expressed as

$$p_D = \sum_{i=1}^{N} \rho_w g \zeta_{a,i} e^{k_i z} \sin(\omega_i t - k_i x + \epsilon_i) \tag{5.65}$$

In general, the excitation forces are derived by integrating the dynamic wave pressure field over the wet surface of the vessel and including diffraction forces. In this case study, only the bottom of the vessel is considered as wetted. The excitation forces in surge, sway and heave are in [142] given as

$$F_i = -\int_S p_D n_{s,i} ds + A_{i,1} a_1 + A_{i,2} a_2 + A_{i,3} a_3 \tag{5.66}$$

where S is the wet surface of the ship, $n_{s,i}$ is an unit vector orthogonal to the surface for an excitation force in the i-direction, $A_{i,j}$ are the added mass terms and a_i is the wave acceleration in the i-direction. Note that when the waves propagate with the x-axis, $a_y = 0$. Expressions for a_x and a_z can be derived based on the wave potential as done in [142]. The excitation torques i roll, pitch and yaw are calculated based on the excitation forces, as will be seen shortly. However, it is not always easy to find algebraic expressions for the final excitation forces and torques. The geometry of the wetted surface is often complex, making it hard to find good integration limits, especially if the heading of the ship is not pointing in the same direction as the propagating waves. This would require a transformation of the integration limits dependent on the wave encounter angle. To avoid both problems, the pressure field can be integrated numerically over the wetted vessel surface. By dividing the wetted surface into small elements it is possible to find approximations of the excitation forces and torques as sums of contributions from each small element. Figure 5.7 shows how the wetted surface, (in this case the bottom area of the vessel), is divided. In the figure β is the angle between the heading of the vessel and the propagating waves. From this division it is more or less straight forward to find an estimate of the excitation forces and torques acting on the ship. When neglecting the end effects, integrating numerically and by working in the body reference frame, the excitation forces and torques from the

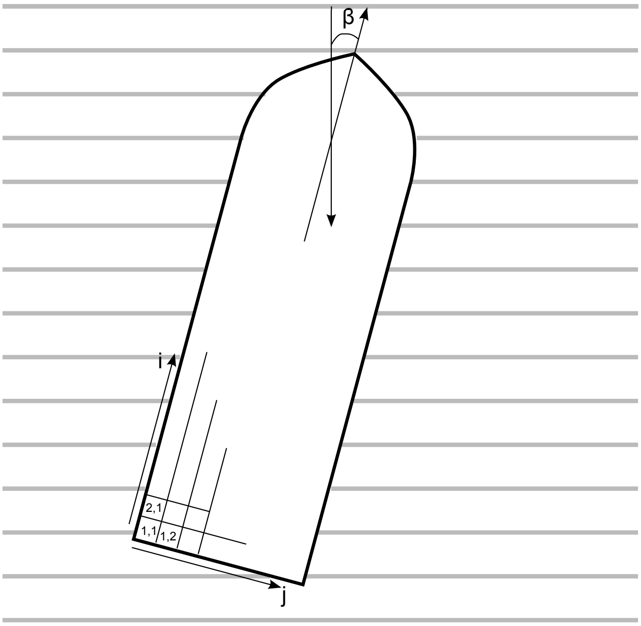


Figure 5.7: Wet surface elements and propagating wave

waves can be expressed as

$$\begin{bmatrix} F_{surge,b,k} \\ F_{sway,b,k} \\ F_{heave,b,k} \end{bmatrix} = \frac{1}{n_i n_j} \begin{bmatrix} X_{\dot{u}} & 0 & 0 \\ 0 & Y_{\dot{v}} & 0 \\ 0 & 0 & Z_{\dot{w}} \end{bmatrix} \boldsymbol{R}_0^b \begin{bmatrix} a_x(x_i, z_i) \\ 0 \\ a_z(x_i, z_i) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ p_D(x_i, z_i)\Delta i \Delta j \end{bmatrix} \tag{5.67a}$$

$$\begin{bmatrix} F_{surge,b} \\ F_{sway,b} \\ F_{heave,b} \end{bmatrix} = \sum_k^K \begin{bmatrix} F_{surge,b,k} \\ F_{sway,b,k} \\ F_{heave,b,k} \end{bmatrix} \tag{5.67b}$$

$$\begin{bmatrix} M_{roll,b} \\ M_{pitch,b} \\ M_{yaw,b} \end{bmatrix} = \sum_k \boldsymbol{r}_k \times \begin{bmatrix} F_{surge,b,k} \\ F_{sway,b,k} \\ F_{heave,b,k} \end{bmatrix} \tag{5.67c}$$

where n_i and n_j are the numbers of wetted elements in the i and j direction, see figure 5.7, the subscript k denotes a given wetted surface component, x_i, y_i and z_i, are the coordinates of each wetted element relative to the inertial reference frame, Δi and Δj are the length and width of each wetted surface element, respectively, $F_{sway,b}$ and $F_{heave,b}$ are the vertical excitation forces given in the body reference frame, $M_{roll,b}$, $M_{pitch,b}$ and $M_{yaw,b}$ are the excitation torques, $\boldsymbol{r}_k = [x_k,\ y_k,\ z_k]^\top$ are the coordinates for the position of each element k related to the pivot centre of the ship. Note that the added mass coefficients are divided by the number of elements the wetted surface were divided into, since they are already given in section 5.5.2. This is only valid if each element is assumed to contribute equally to the total added masses. The excitation forces and torques can be implemented as bond graphs, as shown in figure 5.6, through modulated effort sources, taking input from the integration algorithm.

Since multiple waves are used to form the sea state, second order effects may be included. In addition, second order mean drift forces are included, and given as

$$F_{2d,surge} = P_{surge} \sum_{i=1}^{N} \sum_{j=1}^{N} \zeta_{a,i}\zeta_{a,j} \cos((\omega_i - \omega_j)t + \epsilon_i - \epsilon_j) \sin(\psi) \tag{5.68a}$$

$$F_{2d,sway} = P_{sway} \sum_{i=1}^{N} \sum_{j=1}^{N} \zeta_{a,i}\zeta_{a,j} \cos((\omega_i - \omega_j)t + \epsilon_i - \epsilon_j) \cos(\psi) \tag{5.68b}$$

where

$$P_{surge} = \frac{1}{2}\rho_w g c_{surge} \tag{5.69a}$$

$$P_{sway} = \frac{1}{2}\rho_w g c_{sway} \tag{5.69b}$$

are Newman's approximation coefficients and $c_{surge} \in [0, 1]$ and $c_{sway} \in [0, 1]$ are approximated reflection coefficients. Note that only in-phase slowly varying drift forces are included since out-of-phase slowly varying drift forces are assumed small and negligible in this case study.

5.5.4 Thruster System

The vessel is actuated by two main thrusters in the stern and a tunnel thruster in the bow. The configuration of these are shown in Figure 5.8. The main thrusters generate thrust along the body fixed x-axis, and the tunnel thruster along the body fixed y-axis. In this section, the thruster models are presented. First, the dynamics of an individual thruster is discussed, before the bond graph connections between the thruster system and the bond graph template is presented.

The literature proposes a number of manners in which to model thrusters and propellers, e.g. [143, 144, 145]. In this case study, the two-state thruster model proposed by [146] is used because it is fairly easy to implement, while at the same time including the ambient flow velocity effect. First a motor delivers a torque Q to a propeller shaft, which responds with the angular velocity ω_p. The shaft, motor,

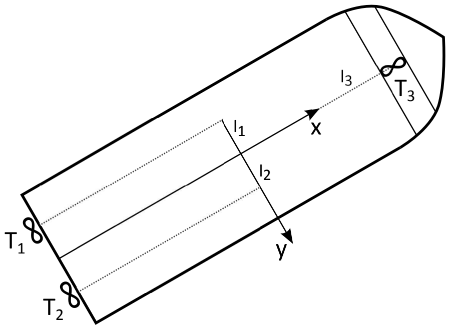


Figure 5.8: Thruster system layout.

and propeller have a moment of inertia denoted by J_p. Furthermore, the total friction force of the propeller shaft bearings and the motor is assumed to be described as $F_p = d_p\omega_p$. The angular velocity of the propeller transforms into a tangential velocity on the propeller blades, $u_p = 0.7R$, according to convention [146], where R is the radius of the propeller. This velocity, together with the incoming velocity u_a due to the flow through the thruster duct combines into the fluid velocity v relative to the propeller blades as shown in Figure 5.9. When a propeller blade propagates through the water with the velocity v relative to the water particles, a lift force L and a drag force D results. These are found as

$$\begin{aligned} L &= \frac{1}{2}\rho_w v^2 A C_L sin(2\alpha) \\ D &= \frac{1}{2}\rho_w v^2 A C_D(1 - cos(2\alpha)) \end{aligned} \tag{5.70}$$

where A is the propeller duct cross section area, C_L and C_D are the lift coefficient and drag coefficient respectively, and α is defined in Figure 5.9. The lift force and drag force can in turn be used to find the thrust force T and the propeller shaft torque Q as

$$\begin{aligned} T &= L\cos(\theta) - D\sin(\theta) \\ Q &= 0.7RF_p = 0.7R(L\sin(\theta) + D\cos(\theta)) \end{aligned} \tag{5.71}$$

where $\theta = p - \alpha$, p is the propeller pitch at $0.7R$, and F_p is the force acting at $0.7R$ on the propeller, resulting in the torque Q.

The thrust force acts to accelerate the fluid in the propeller duct and to create a friction force between the water and the thruster duct. The relative velocity between the thruster duct and the water is $\bar{u}_a = u_a - u_T$, where u_T is the velocity of the duct. The sum of the inertial force related to the acceleration of the fluid,

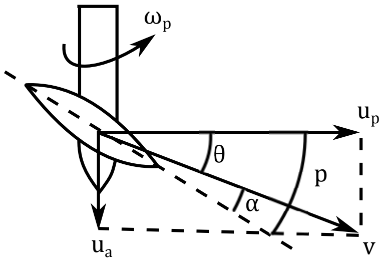


Figure 5.9: Propeller velocities.

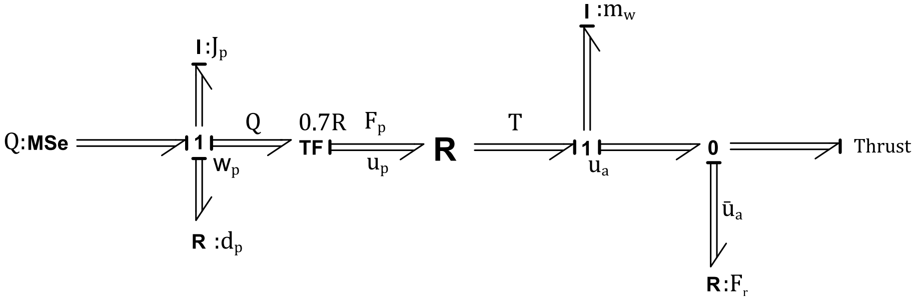


Figure 5.10: Bond graph of thruster dynamics model.

and the friction force must be equal to the thrust force. Thus, we have

$$m_w \dot{u}_a + F_r(\bar{u}_a) = T \tag{5.72}$$

where m_w is the mass of the fluid in the duct and $F_r(\bar{u}_a)$ is the friction force. In this case study, it is assumed that the friction force is described as

$$F_r(\bar{u}_a) = 2\rho_w A|\bar{u}_a|\bar{u}_a \tag{5.73}$$

The above equations are described by the bond graph shown in Figure 5.10, where all three thrusters are represented in the vector formulation. Note that the motors driving the propeller shafts are modelled simply as effort sources here, but in Chapter 8 a power plant including propulsor drives are included. For increased model fidelity, these effort sources could be replaced by variable frequency drive models, in turn powered by e.g., a diesel electric power system.

The forces and torques on the vessel due to the thrusters act in the body fixed reference frame as the thrusters are fixed to the body in this case study. As such, the thruster forces should be interfaced to the 1-junction representing $\boldsymbol{v}^b_{b/0}$, and the torques to the 1-junction representing $\boldsymbol{\omega}^b_{b/0}$. Then, it remains to make connections between the individual thrust forces and the resulting forces and torques on the vessel. Consider the vector $\boldsymbol{T} = [T_1, T_2, T_3]^\top$, representing the individual thrust forces from each thruster, and the vector $\boldsymbol{\tau}_T^b = [(\boldsymbol{\tau}_{Tv}^b)^\top, (\boldsymbol{\tau}_{T\omega}^b)^\top]^\top$, where $\boldsymbol{\tau}_{Tv}^b$ and $\boldsymbol{\tau}_{T\omega}^b$ are the 3×1 vectors of resulting thrust forces and thrust torques on the vessel respectively. Letting l_1 and l_2 be the distances in y_b-direction from the origin of the body fixed reference frame to the first and second thrusters, and l_3, the distance in x_b direction to the third thruster, and finally, l_4, l_5, and l_6, the distances in z_b-direction to the first, second and third thruster, we find that

$$\begin{aligned}
\boldsymbol{\tau}_{Tv}^b &= \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{T} = \boldsymbol{H}_v \boldsymbol{T} \\
\boldsymbol{\tau}_{T\omega}^b &= \begin{bmatrix} 0 & 0 & l_6 \\ l_4 & l_5 & 0 \\ l_1 & -l_2 & l_3 \end{bmatrix} \boldsymbol{T} = \boldsymbol{H}_\omega \boldsymbol{T}
\end{aligned} \tag{5.74}$$

Consider again the thruster velocity component u_{Ti}, this time specified for the i-th thruster, in the direction in which the thruster in question produces thrust. There are two contributions to this velocity component; the contribution $u_{Ti}^{(v)}$ from the linear velocity of the vessel, $\boldsymbol{v}_{b/0}^b$, and the contribution $u_{Ti}^{(\omega)}$ from the angular velocity of the vessel, $\boldsymbol{\omega}_{b/0}^b$. Collecting the velocity components for each of the three thrusters in to a 3×1 vector, we can write $\boldsymbol{u}_T = \boldsymbol{u}_T^{(v)} + \boldsymbol{u}_T^{(\omega)}$. The contribution from the linear velocity and the angular velocity of the vessel can be found as

$$\begin{aligned}
\boldsymbol{u}_T^{(v)} &= (\boldsymbol{H}_v)^\top \boldsymbol{v}_{b/0}^b \\
\boldsymbol{u}_T^{(\omega)} &= (\boldsymbol{H}_\omega)^\top \boldsymbol{\omega}_{b/0}^b
\end{aligned} \tag{5.75}$$

respectively. Thus, the thruster system and the vessel can be connected as shown in Figure 5.6. Note that later on, in Chapter 7, a thrust allocation algorithm for rotatable thrusters will be derived. In the next section, control of the thrusters are provided through a DP-control system.

5.5.5 DP-control system

The objective of the DP-control system is to provide reference signals for the thrusters such that the vessel is controlled in surge, sway and yaw. The DP-system consists of a second order reference model, smoothing the position and yaw angle set point into a reference signal, a position controller calculating a desired thrust vector, a thrust allocation algorithm using the desired thrust vector to allocate a desired thrust force for each thruster, local thruster controllers that realize the thrust commands, and a non-linear passive observer in order to filter out high frequency

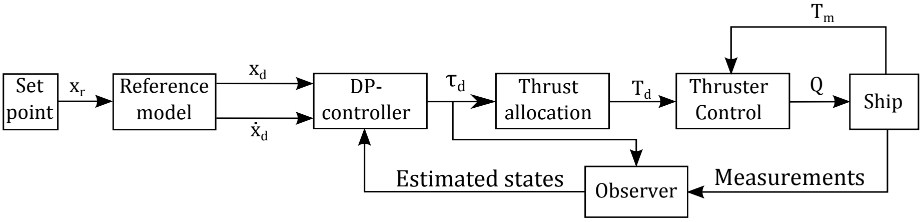


Figure 5.11: Ship control system layout.

components of the position and angle measurements in addition to the corresponding rates. The overall control system layout for the ship is illustrated in Figure 5.11. A detailed survey of DP-control systems and different state-of-the-art techniques for the different subsystems of the DP-control system can be found in [20].

The reference model is implemented as a second order filter with velocity saturation, which takes a reference position, $\boldsymbol{x}_r$, and provides the filtered position $\boldsymbol{x}_d$ and the velocity $\dot{\boldsymbol{x}}_d$ as input to the DP-controller. More details of this implementation can be found in [14]. The DP-controller then compares these reference states to corresponding states from the vessel observer, employing a PID control-law that calculates a desired thrust vector in terms of the inertial reference frame. This can be transformed into the body-fixed frame by using the rotation matrix $\boldsymbol{R}_0^b$. However, in the control system, we are only concerned with the position in the horizontal plane and the yaw angle, and therefore the transformation is simplified to $\boldsymbol{R}_0^b(\phi = 0, \theta = 0, \psi) = (\boldsymbol{R}_z(\psi))^\top$.

The control error is given as

$$\boldsymbol{e} = \boldsymbol{x}_d - \hat{\boldsymbol{x}} \tag{5.76}$$

where $\hat{\boldsymbol{x}}$ is the position and yaw angle estimates. The derivative of the error is given as

$$\boldsymbol{e}_D = \dot{\boldsymbol{x}}_d - \dot{\hat{\boldsymbol{x}}} \tag{5.77}$$

and the integral of the error is given as

$$\boldsymbol{e}_I = \int_{\tau=0}^{t} \boldsymbol{e} d\tau \tag{5.78}$$

The control forces $\boldsymbol{\tau}_c^0$ given in the inertial reference frame is then

$$\boldsymbol{\tau}_c^0 = \boldsymbol{K}_p^{ship}\boldsymbol{e} + \boldsymbol{K}_D^{ship}\boldsymbol{e}_D + \boldsymbol{K}_I^{ship}\boldsymbol{e}_I \tag{5.79}$$

where $\boldsymbol{K}_p^{ship}$, $\boldsymbol{K}_D^{ship}$ and $\boldsymbol{K}_I^{ship}$ are the control gain matrices. The control forces given in the body reference frame is

$$\boldsymbol{\tau}_c^b = (\boldsymbol{R}_z(\psi))^\top \boldsymbol{\tau}_c^0 \tag{5.80}$$

This thrust vector command is provided as input to the thrust allocation in order to find a thrust force reference for each thruster. A survey of different methods for thrust allocation is provided in [147]. In this case-study, however, the problem is rather trivial because the vessel is not over-actuated, (i.e., there are three thrusters and three degrees of freedom that we seek to control). As such, the problem can be solved by multiplying the thrust command by a reduced version, $\bar{\boldsymbol{H}}$, of the thrust allocation matrix $[(\boldsymbol{H}_v)^\top, (\boldsymbol{H}_\omega)^\top]^\top$, where only the relevant degrees of freedom for control are extracted. This gives the relation

$$\boldsymbol{\tau}_c^b = \bar{\boldsymbol{H}}\boldsymbol{T}_c = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ l_1 & -l_2 & l_3 \end{bmatrix} \boldsymbol{T}_c \tag{5.81}$$

The thrust can then be allocated by inverting (5.81).

The thrust command for each thruster is realized by local thruster controllers. These are controllers that in reality are hard to design because one generally does not have access to measurements of the thrust force. A number of references to recent literature on the local thruster control problem is provided in [20]. In this chapter however, we have assumed that we have access to perfect measurements of the thrust. It is furthermore assumed that the electrical motors driving the propeller shaft follows perfectly a desired torque, $\boldsymbol{Q}_d$. Doing so, we can define the error between the commanded thrust and the measured thrust as

$$\boldsymbol{e}_T = \boldsymbol{T}_d - \boldsymbol{T}_m \tag{5.82}$$

and then set the desired motor torques as

$$\boldsymbol{Q}_d = \boldsymbol{K}_{pT}\boldsymbol{e}_T + \boldsymbol{K}_{iT}\int_0^t \boldsymbol{e}_T dt \tag{5.83}$$

The actual motor torque is given as

$$\boldsymbol{Q} = \text{sat}(\boldsymbol{Q}_d, -\boldsymbol{Q}_{\text{lim}}, \boldsymbol{Q}_{\text{lim}}) \tag{5.84}$$

where

$$\boldsymbol{Q}_{\text{lim}} = \frac{\boldsymbol{P}_{\text{max}}}{\boldsymbol{\omega}_p} \tag{5.85}$$

and $\boldsymbol{P}_{\text{max}}$ is the maximum power rating vector and $\boldsymbol{\omega}_p$ is the speed vector for the thrusters. Thus, $\boldsymbol{Q}_{\text{lim}}$ is the vector of torque limits for the thrusters. As the thrusters can be saturated, integrator anti-wind up algorithms are implemented for the controllers described in (5.79) and (5.83).

The non-linear passive observer (NLPO) is implemented according to [148]. The purpose of this observer is to estimate the high frequency components of the position and angle measurements, i.e., wave frequency and higher, and make sure they do not enter the controller feedback loop. If this is not done, the controller will seek to compensate for the motion induced by the first order wave forces. This however would require immense amounts of power, and is not desirable. Rather, the control system should compensate for only the slowly varying and constant disturbances such that the ship is free to oscillate with the wave frequency. This observer generates estimates of the north and east position, the yaw angle, the low frequency velocities and the bias force. This is achieved by running a simplified dynamic model, and then correcting for the difference in measurements and estimates, i.e., the estimation error. The dynamic model used is given in [148] as

$$\dot{\hat{\boldsymbol{\xi}}} = \boldsymbol{A}_\omega \hat{\boldsymbol{\xi}} + \boldsymbol{K}_1 \tilde{\boldsymbol{y}} \tag{5.86a}$$

$$\dot{\hat{\boldsymbol{r}}}^0_{b/0} = \boldsymbol{R}_z(\psi)\hat{\boldsymbol{v}}^b_{b/0} + \boldsymbol{K}_2 \tilde{\boldsymbol{y}} \tag{5.86b}$$

$$\dot{\hat{\boldsymbol{b}}} = -\boldsymbol{T}^{-1}\hat{\boldsymbol{b}} + \boldsymbol{K}_3 \tilde{\boldsymbol{y}} \tag{5.86c}$$

$$\boldsymbol{M}\dot{\hat{\boldsymbol{v}}}^b_{b/0} = -\boldsymbol{D}\hat{\boldsymbol{v}}^b_{b/0} + (\boldsymbol{R}_z(\psi))^\top \hat{\boldsymbol{b}} + \boldsymbol{\tau}^b_c + \boldsymbol{K}_4 \tilde{\boldsymbol{y}} \tag{5.86d}$$

$$\tilde{\boldsymbol{y}} = \hat{\boldsymbol{y}} + \boldsymbol{C}_\omega \hat{\boldsymbol{\xi}} \tag{5.86e}$$

where $\hat{\boldsymbol{\xi}}$ is the wave response estimate on position and heading, $\hat{\boldsymbol{y}}$ is the position and heading estimate, $\tilde{\boldsymbol{y}} = \boldsymbol{r}^0_{b/0} - \hat{\boldsymbol{y}}$ is the position estimation error, $\hat{\boldsymbol{r}}^0_{b/0}$ and $\hat{\boldsymbol{v}}^b_{b/0}$ are the estimates on the states $\boldsymbol{r}^0_{b/0}$ and $\boldsymbol{v}^b_{b/0}$. The matrices $\boldsymbol{K}_1 \in \mathcal{R}^{6\times3}$, $\boldsymbol{K}_2 \in \mathcal{R}^{3\times3}$, $\boldsymbol{K}_3 \in \mathcal{R}^{3\times3}$, and $\boldsymbol{K}_4 \in \mathcal{R}^{3\times3}$ are tuning parameters. The bias force estimate is denoted $\hat{\boldsymbol{b}}$, and $\boldsymbol{T}$ is the time constant matrix of the bias force, and can also be considered as a tuning parameter. (5.86a) is a state space representation of the motion component of the ship due to wave forces, driven by the estimation error. $\boldsymbol{M}$ is the mass matrix of the ship, and $\boldsymbol{D}$ is in general the linear damping matrix. This damping matrix has been modified to also include nonlinear damping terms, such as $\boldsymbol{D}(\boldsymbol{v}^b_{b/0})$. The linear wave spectra is characterized by the matrix $\boldsymbol{A}_w$ given as

$$\boldsymbol{A}_w = \begin{bmatrix} \boldsymbol{0}_{3\times3} & \boldsymbol{I}_{3\times3} \\ -\mathrm{diag}(\omega_{0i}^2) & -2\mathrm{diag}(\lambda_i\omega_{0i}) \end{bmatrix} \tag{5.87}$$

where ω_{0i} is the peak frequency in the wave spectra, and λ_i is a spectra tuning parameter. Finally, the matrix $\boldsymbol{C}_\omega = [\boldsymbol{0}_{3\times3}, \boldsymbol{I}_{3\times3}]$, such that the ship motions induced by the linear wave model is extracted.

5.5.6 Wire model

In [125, 149] a lumped wire model in two degrees of freedom is presented. The wire is divided into smaller elements and connected like a mass spring damper system in series, as Figure 5.12 shows.

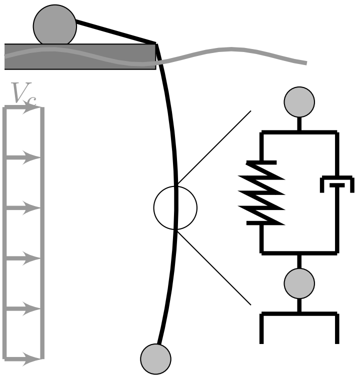


Figure 5.12: Sketch of winch in operation.

The same idea is used here and the only difference is that the wire model is updated to have three degrees of freedom in each node instead of two. The wire is also assumed to be directly connected to the crane and is not to be hoisted or lowered in this case study.

Starting with the wire dynamics, the stiffness of each wire element is given in [125] as

$$k_w = \frac{EA_w}{L_{we}} = \frac{ED_w^2\pi}{4L_{we}} \tag{5.88}$$

where E is the elasticity modulus, A_w is the cross section area of the wire, L_{we} is the length of each wire element and D_w is the diameter of the wire. The damping is found by assuming a constant damping ratio,

$$\zeta = \frac{c_{wire}}{c_{cr}} \tag{5.89}$$

where c_{wire} is the damping coefficient, c_{cr} is the critical damping coefficient and ζ is the damping ratio. The wire is assumed to be over-damped, which means that $\zeta >> 1$. The critical damping coefficient is given as

$$c_{cr} = 2m_{we}\sqrt{\frac{k_w}{m_{we}}} = 2\sqrt{k_w m_{we}} \tag{5.90}$$

where m_{we} is the translational inertia of the wire element in air,

$$m_{we} = \rho_{wire} A_w L_{we} \tag{5.91}$$

where ρ_{wire} is the density of the wire material. By inserting (5.90) into (5.89) the damping coefficient can be expressed as

$$c_{wire} = 2\zeta\sqrt{k_w m_{we}} \tag{5.92}$$

The elongation of one wire element is given as

$$r_{we} = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2} - L_0 \tag{5.93}$$

where (x_0, y_0, z_0) and (x_1, y_1, z_1) is the position of the top and bottom of the wire element, respectively, and L_0 is the initial length of the unstretched wire element. The derivative is given as

$$\begin{aligned}\dot{r}_{we} &= \frac{(x_1 - x_0)\dot{x}_1 + (y_1 - y_0)\dot{y}_1 + (z_1 - z_0)\dot{z}_1}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}} \\ &- \frac{(x_1 - x_0)\dot{x}_0 + (y_1 - y_0)\dot{y}_0 + (z_1 - z_0)\dot{z}_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}} \\ &= (\dot{x}_1 - \dot{x}_0)r_{t1} + (\dot{y}_1 - \dot{y}_0)r_{t2} + (\dot{z}_1 - \dot{z}_0)r_{t3}\end{aligned} \tag{5.94}$$

where

$$r_{t1} = \frac{x_1 - x_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}} \tag{5.95a}$$

$$r_{t2} = \frac{y_1 - y_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}} \tag{5.95b}$$

$$r_{t3} = \frac{z_1 - z_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}} \tag{5.95c}$$

and is the tranformer modulus between the rates in three degrees of freedom and the rate of the wire elongation. By using Morrison's equation both the added mass and the drag forces can be found,

$$\begin{aligned}\boldsymbol{F_M} &= \rho_w A_w L_{we} \begin{bmatrix} C_{I,x}\ddot{x} \\ C_{I,y}\ddot{y} \\ C_{I,z}\ddot{z} \end{bmatrix} \\ &+ \frac{1}{2}\rho_w D_w L_{we} \begin{bmatrix} C_{d,x}\cos(\theta_1)(\dot{x} - V_{c,x})|\dot{x} - V_{c,x}| \\ C_{d,y}\cos(\theta_2)(\dot{y} - V_{c,y})|\dot{y} - V_{c,y}| \\ C_{d,z}\sin(\theta_3)(\dot{z} - V_{c,z})|\dot{z} - V_{c,z}| \end{bmatrix}\end{aligned} \tag{5.96}$$

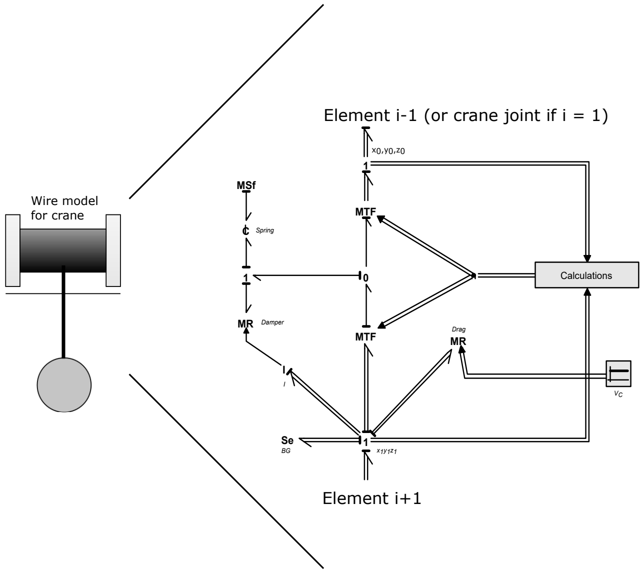


Figure 5.13: Bond graph model of one wire element

where $C_{I,i}$ is the added mass coefficients, $C_{d,i}$ is the drag coefficients, $V_{c,i}$ is the current velocity in each direction, and

$$\theta_1 = -\frac{y_1 - y_0}{z_1 - z_0} \tag{5.97a}$$

$$\theta_2 = -\frac{x_1 - x_0}{z_1 - z_0} \tag{5.97b}$$

$$\theta_3 = -\frac{\sqrt{x_1^2 + y_1^2} - \sqrt{x_0^2 + y_0^2}}{z_1 - z_0} \tag{5.97c}$$

The last contribution to the wire element dynamics are the buoyancy and the gravitational forces, which is given as

$$\boldsymbol{F}_{BG} = [0, 0, (m_{we} - \rho_w A_w L_{we})g]^\top \tag{5.98}$$

By implementing these equation in bond graphs, one wire element can be given as Figure 5.13 shows. In addition the first and last wire elements must be slightly modified. The first wire element must have a connection to the tip of the crane, and the last wire element must have the payload characteristics included in the mass and the drag forces.

5.5.7 Crane Control System and Actuators

The crane is equipped with hydraulic actuators. In this case study however, these are simplified as effort sources, providing a torque which in turn is commanded from a crane joint position control system. The manner in which the effort sources are connected to the model is shown in Figure 5.14. Notice that a reaction torque is acting on the second link due to the actuator on the third joint. This is because the second and third link both rotate about the horizontal axis. Also here a reference model for generating a smooth and continuous reference signal is needed, and a similar filter as for the reference model of the vessel is used. Hence, the reference model is given as

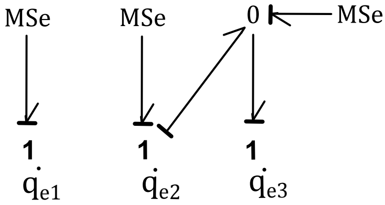


Figure 5.14: Bond graph of crane actuator system.

$$
\begin{aligned}
&\boldsymbol{M}_{ref}^{crane}\ddot{\boldsymbol{x}}_d^c + \boldsymbol{D}_{ref}^{crane}\dot{\boldsymbol{x}}_d^c + \boldsymbol{K}_{ref}^{crane}\boldsymbol{x}_d^c = \boldsymbol{x}_r^c \\
&\dot{\boldsymbol{x}}_d^c = \text{sat}(\dot{\boldsymbol{x}}_{min}^c, \dot{\boldsymbol{x}}_{max}^c)
\end{aligned}
\tag{5.99}
$$

where $\boldsymbol{M}_{ref}^{crane}$, $\boldsymbol{D}_{ref}^{crane}$ and $\boldsymbol{K}_{ref}^{crane}$ are tuning matrices for the reference model, $\dot{\boldsymbol{x}}_{min}^c$ and $\dot{\boldsymbol{x}}_{max}^c$ are the angular rate limits and $\boldsymbol{x}_r^c$ is the reference signals given as input to the filter.

The control law for the crane is a Lyapunov stability based control design that enables cancellation of unwanted crane dynamics. The isolated crane dynamics, (i.e., the dynamics of the crane if we do not include the vessel), can, according to e.g. [150], be expressed as

$$
\begin{aligned}
\dot{\boldsymbol{q}}_c &= \boldsymbol{\omega}_c \\
\dot{\boldsymbol{\omega}}_c &= \boldsymbol{B}_c^{-1}(\boldsymbol{q}_c)(-\boldsymbol{C}_c(\boldsymbol{q}_c, \boldsymbol{\omega}_c)\boldsymbol{\omega}_c - \boldsymbol{g}_c(\boldsymbol{q}_c) + \boldsymbol{\tau}_c)
\end{aligned}
\tag{5.100}
$$

By defining the control error vector as

$$
\boldsymbol{e}_{c1} = \boldsymbol{q}_c - \boldsymbol{q}_{cd} \tag{5.101}
$$

where $\boldsymbol{q}_c$ is the angle measurement vector and $\boldsymbol{q}_{cd}$ is the reference angle vector, and

$$
\boldsymbol{e}_{c2} = \boldsymbol{\omega}_c - \dot{\boldsymbol{q}}_{cd} \tag{5.102}
$$

where $\boldsymbol{\omega}_c$ is the angular rate vector and $\dot{\boldsymbol{q}}_{cd}$ is the reference angular vector, it is possible to write the error dynamics as

$$
\begin{aligned}
\dot{\boldsymbol{e}}_{c1} &= \boldsymbol{e}_{c2} \\
\dot{\boldsymbol{e}}_{c2} &= \boldsymbol{B}_c^{-1}(\boldsymbol{q}_c)(-\boldsymbol{C}_c(\boldsymbol{q}_c, \boldsymbol{\omega}_c)\boldsymbol{\omega}_c - \boldsymbol{g}_c(\boldsymbol{q}_c) + \boldsymbol{\tau}_c) - \ddot{\boldsymbol{q}}_{cd}
\end{aligned}
\tag{5.103}
$$

Starting with choosing a Lyapunov function candidate given as

$$V = \frac{1}{2}\boldsymbol{e_2}^\top \boldsymbol{B}(\boldsymbol{q}_c)\boldsymbol{e_2} \tag{5.104}$$

which is positive definite for $\forall\, \boldsymbol{e}_2 \neq 0$, gives

$$\dot{V} = \boldsymbol{e}_2^\top(-\boldsymbol{C}_c(\boldsymbol{q}_c, \boldsymbol{\omega_c})\boldsymbol{\omega}_c - \boldsymbol{g}_c(\boldsymbol{q}_c) + \boldsymbol{\tau}_c - \boldsymbol{B}(\boldsymbol{q}_c)\ddot{\boldsymbol{q}}_{cd}) \tag{5.105}$$

By choosing

$$\boldsymbol{\tau}_c = \boldsymbol{C}_c(\boldsymbol{q}_c, \boldsymbol{\omega_c})\boldsymbol{\omega}_c + \boldsymbol{g}_c(\boldsymbol{q}_c) + \boldsymbol{B}(\boldsymbol{q}_c)\ddot{\boldsymbol{q}}_{cd} - \boldsymbol{K}_d\boldsymbol{e}_2 + u \tag{5.106}$$

the derivative of the Lyapunov function becomes

$$\dot{V} = -\boldsymbol{e}_2^\top \boldsymbol{K}_d\boldsymbol{e}_2 + \boldsymbol{e}_2^\top u \tag{5.107}$$

where $\boldsymbol{K}_d$ is a positive diagonal matrix. This would stabilize $\boldsymbol{e}_2$ when $u = 0$, and the new error dynamics become

$$\begin{aligned} \dot{\boldsymbol{e}}_{c1} &= \boldsymbol{e}_{c2} \\ \boldsymbol{B}(\boldsymbol{q}_c)\dot{\boldsymbol{e}}_{c2} &= -\boldsymbol{K}_d\boldsymbol{e}_2 + u \end{aligned} \tag{5.108}$$

Since it is known that $\boldsymbol{e}_2$ is stabilized from the previous Lyapunov function, it is reasonable to believe that u can be chosen as a PI-controller that stabilizes $\boldsymbol{e}_1$,

$$u = -\boldsymbol{K}_p\boldsymbol{e}_1 - \boldsymbol{K}_I \int_0^t \boldsymbol{e}_1 dt \tag{5.109}$$

This assumption should be verified through simulations. Hence, the total control law is then given as

$$\begin{aligned} \boldsymbol{\tau}_c = {} & \boldsymbol{C}_c(\boldsymbol{q}_c, \boldsymbol{\omega_c})\boldsymbol{\omega}_c + \boldsymbol{g}_c(\boldsymbol{q}_c) + \boldsymbol{B}(\boldsymbol{q}_c)\ddot{\boldsymbol{q}}_{cd} \\ & - \boldsymbol{K}_d\boldsymbol{e}_2 - \boldsymbol{K}_p\boldsymbol{e}_1 - \boldsymbol{K}_I \int_0^t \boldsymbol{e}_1 dt \end{aligned} \tag{5.110}$$

This case study has been implemented as seen in Figure 5.6 and simulation results comparing with and without crane load are shown in the following section.

5.6 Simulation Results

In this chapter the importance of proper modelling of heavy deck equipment tightly coupled to the vessel in simulation models for maritime operations have been studied. In this section, we present simulation results from the case-study model derived in the previous section to emphasize the main results and to illustrate the

Table 5.2: Simulation Parameters.

Parameter	**Value**
Vessel	
Initial position and orientation	$[0\,\mathrm{m}, 0\,\mathrm{m}, -25\,°]$
Power saturation main thrusters	3.5 MW
Power saturation tunnel thruster	3.5 MW
Sea state parameters	
Significant wave height	1.5 m
Peak period	8 s
Northward current	-0.2 m/s
No. wave components	50 -
γ	3 -
Lower wave spectra period	0.2 s
Upper wave spectra period	50 s
Δi	10.7 m
Δj	4.4 m
Crane and submerged load model	
Initial and reference joint angles	$[90\,°, 30\,°, -30\,°]$
Wire diameter	0.05 m
Initial wire length	400 m
No. wire elements	5 -
Payload	100 tons

importance of proper modeling. In particular, the same DP-manoeuvre, shown in Figure 5.15 and 5.16, is performed both with and without a submerged load attached to the crane. In both cases, the reference position is first moved 80 meters northwards from the initial position, while the east -and yaw reference is held constant. Then the reference position is moved 20 meters eastwards while the yaw reference still is held constant at -25 ° before finally, the yaw reference is changed to 25 °. Note that all controllers and filters have been tuned to perform well in the case with no load attached to the crane, and the same tuning is used for the case with loaded crane. The model parameters used are presented in Table 5.2.

In such a case study it is important that the wave filter and the control systems are tuned to be robust such that the vessel is able to perform good in both cases and keep its reference position and orientation. If this is achieved, there should not be large differences in the position and orientation when comparing the two cases, and it is believed that the second order mean drift wave forces are much larger than the environmental forces from the submerged wire and the load in this case study, resulting in small differences in power consumption as well. If this is the case, the

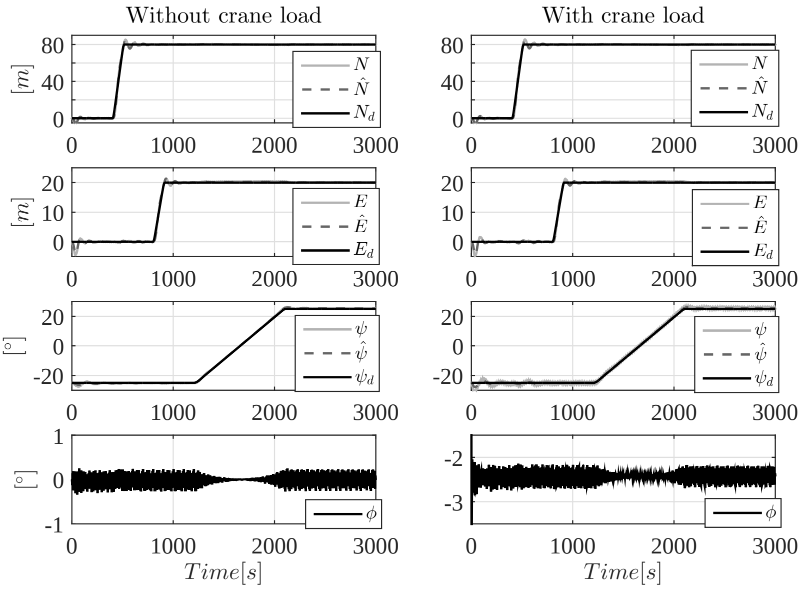


Figure 5.15: Actual, estimated and reference position, yaw- and roll angle of the DP-vessel for the two simulation cases.

main differences between the two cases would be reflected in the roll angle and in the wave filter.

Figure 5.15 shows that the vessel follows its references signals in both cases, and the results seem to overlap. The crane load also cause a static roll angle offset of about -2.5 °, having oscillations with an amplitude of about 0.5 °, about the same amplitude as in the unloaded crane case. Notice that these roll oscillations are significantly reduced when the heading is 0 °. and the reduction is largest in the unloaded crane case due to no static roll angle offset. From this figure one can conclude that the implemented controllers and filters introduced in section 5.5 seem to perform well. Notice also that the submerged crane load have a considerable effect on the yaw angle oscillations, ψ, as can be seen from the actual yaw rate in Figure 5.16. One of the reasons for this is that the force acting on the vessel, due to the submerged load, create a torque about the body fixed z-axis because the crane is situated aft of the center of gravity. These results also argue for the use of proper rigid body models when testing control systems and tuning wave filters for vessels doing crane operations. The figure indicates that the filtered velocities and the yaw rate coincide with the measurements.

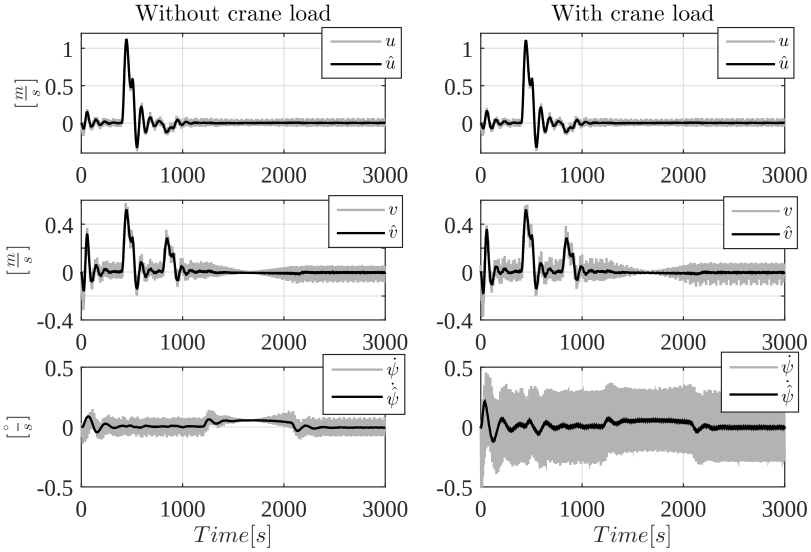


Figure 5.16: Actual and estimated velocities and yaw rates for the two cases.

Figure 5.16 shows the filtered position- and orientation rates, that are fed to the DP-controller for calculating the derivative controller effects, compared with the actual rates. As can be seen in the figure the surge rates u for the two different cases seem to converge and the wave filter filters out about the same amount of noise in the two cases. However, the same can not be said about the sway rate and the heading rate. The noise in the sway rate has a bit larger amplitude in the loaded case compared to the unloaded case, but it seems like the wave filter is able to perform equally in both cases. The last two plots show the heading rate in the two different cases. As can be seen, the noise in the heading rate for the loaded case is significantly increased in comparison to the unloaded case. This, due to the submerged wire and the load. However, even though the wave filter is not able to filter out as much noise in the loaded case in comparison to the unloaded case, the performance of the filter is good and it filters out most of it. Due to these results, it is expected that at least the power consumption for the tunnel thruster, P_3, would contain more noise in the loaded case in comparison to the unloaded case.

The power consumption for the two cases is shown in Figure 5.17, where the the power consumption of each thruster, as well as the total power consumption of the thruster system and the total amount of consumed energy are shown. It is clear from this figure that the submerged load does not increase the energy consumption considerably, but affects the dynamics of the vessel, as will be shown later on. The

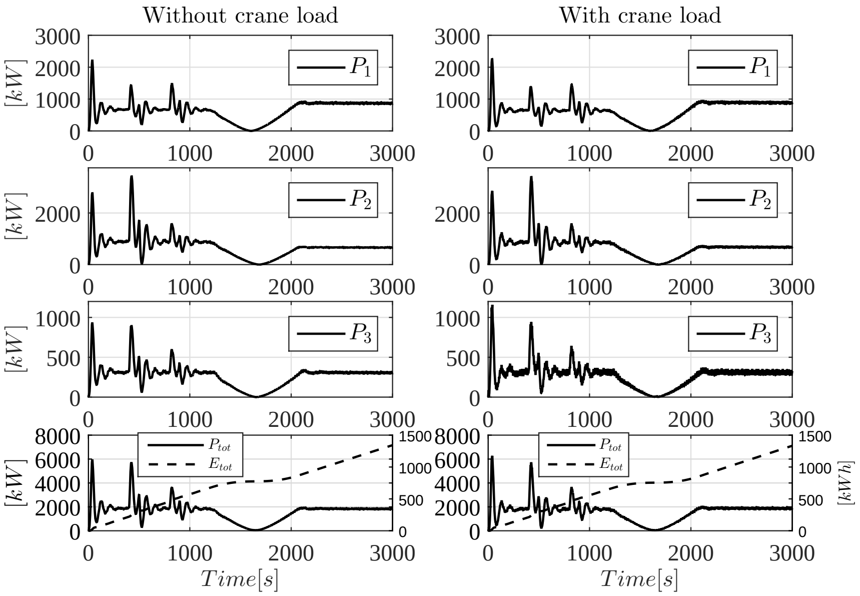


Figure 5.17: Thruster system power consumption for the two cases.

figure also shows that the largest power peaks come from moving the ship and changing the heading. The case with the loaded crane consumes about 9.3 kWh more than the case without loaded crane, i.e., about 0.7 % difference, and about 1333.5 kWh in total. As expected, there are more oscillations in consumed power in the loaded case, especially for the tunnel thruster, in comparison to the unloaded case, due to the resulting forces from the submerged wire and load. However, the results show that the control laws and the wave filter preform well in both cases, which also tell us that both the control laws and the wave filter have been successfully tuned to be robust. Note that the power consumption for the crane system has not been included. It would be interesting in further work to look at energy regenerative control of the crane system in heave compensation operations, which this model is well suited for.

To be able to pinpoint the differences in the simulation results even better the difference in crane joint angles, ship position and yaw- and roll angles are compared in Figure 5.18. As can be seen in the figure there is a small oscillating difference between the crane joint angles, as one would expect. Note that the crane joint angle measurements for the crane control system have not been filtered before entering the control system. However, the largest differences can be seen in the ship position and orientations. The difference in heading oscillates with an amplitude

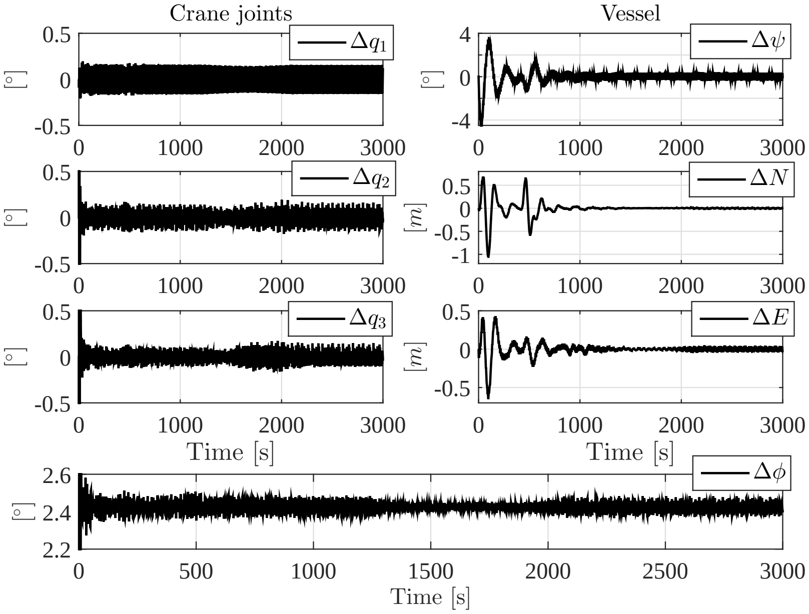


Figure 5.18: Difference in crane joint angles and vessel position and yaw angle in the two simulations.

of about 1 °, indicating that the loaded crane affects the heading of the ship significantly. The same can be said about the north-east position of the ship and the roll angle, as seen in the figure, although not as dramatically for the north-east position. The comparison between the roll angles for the two cases show that the loaded case gets an offset of about 2.5 ° as mentioned earlier. However, this is not the only result that can be obtained from the roll angle measurements. The roll angle is not controlled in any sense in the simulation, and is therefore not directly affected by the control law dynamics. This makes these measurements important when analysing the differences in the dynamics due to the submerged wire and the load. Figure 5.19 shows a FFT-analysis of the roll angle in the time range $t = [2500, 3000]$ for the two cases. As can be seen in the figure the responses seem to be equal for low frequencies. However, when the is in the range 0.1-0.5 Hz the results show that the roll angle for the unloaded vessel is more affected by the wave effects. This is not surprising since the submerged load in the loaded case acts as a mooring line and adds additional damping to the roll angle. However, the three major peaks that can be seen in the unloaded case around 0.12 Hz, 0.17 Hz and 0.27 Hz can also be seen in the loaded case. These results also argues for that the submerged wire and load dynamics add significant contributions to the vessel dynamics, especially

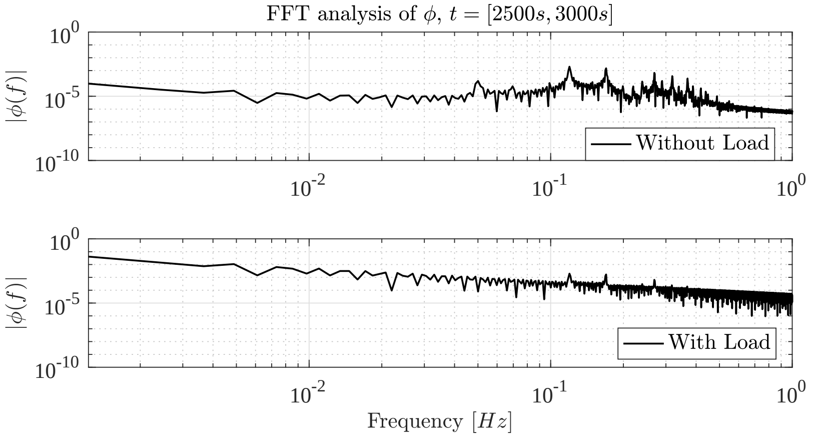


Figure 5.19: Fast Fourier Transform analysis of roll angle for both simulation cases.

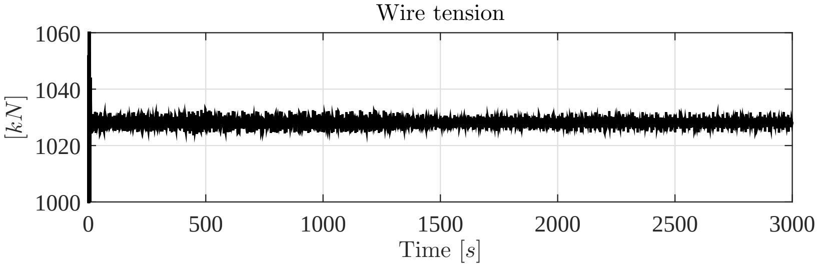


Figure 5.20: Wire tension.

the roll dynamics that are uncontrolled.

It is not surprising that the submerged payload has such an impact on the motion of the vessel, which is also argued for by looking at the wire tension in Figure 5.20, showing that the wire tension oscillates about 1028 kN with an amplitude of about 5 kN. One could perhaps expect that the wire tension would change more due to the change in north- and east position, but the ship moves quite slowly and does not affect the wire tension considerably. However, if the ship had moved faster the wire tension would have changed. Also note that the oscillations are lower when the heading of the ship is 25 ° compared to when the heading is -25 °. This has to do with the orientation of the crane, which affects the roll angle more when the ship has a heading of 25 °, and since the roll is not controlled, the oscillations in the wire tension would decrease as seen in the figure.

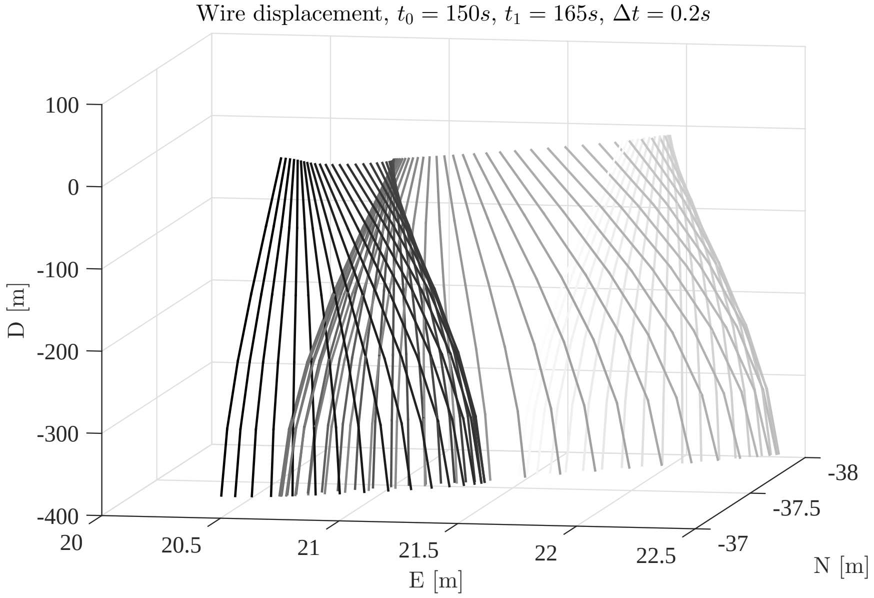


Figure 5.21: Wire displacement.

Figure 5.21 shows the displacement of the submerged wire for a period of 15 seconds. Note that darker color denotes higher simulation time. The figure shows the wire displacement in the time interval when the vessel is moving northwards.

5.7 Conclusion

In this chapter we set out to make a template for developing simulation models of ships doing heavy lift operations using cranes. In order to do this, we formulated the dynamics of the marine vehicle in a compact manner, which allowed for connecting equipment such as cranes and manipulators in a true manner. If this were to be modelled directly without the Lagrangian formalism, challenges related to differential causality would arise. These are solved automatically when developing the Lagrangian equations. Alternative approaches for resolving the differential causalities are to employ so-called brute force techniques, which in general means to introduce some compliance between the rigid bodies. This does however introduce fast time constants which would affect the simulation time significantly. It should be mentioned that the cases presented in this chapter were both solved faster than real time. Note however that the simulation speed is affected by the mesh size of the wetted surface in the numeric integration of the wave forces, and the number of wave components used to describe the irregular sea.

In the development of the Lagrangian equations of motion, the associated potential energy and the conservative forces were not included, but rather modeled directly in the bond graph implementation. The case study further illustrated how potential forces, as well as added mass, can be included in the bond graph model without going through the Lagrangian formalism.

Suggestions for how to interface a variety of different subsystems such as environmental forces, thruster models, wire-load model and crane actuators, were presented through a case study. In addition, a DP-control system, a non-linear wave filter, and a crane control system were implemented. The purpose of the control systems were to enable comparison of the two cases simulated. In addition, the non-linear passive wave filter was included in order to get a realistic power consumption.

The simulation results indicated that the crane and wire-load model affected the performance of the ship. However, since the control laws and the wave filter were tuned to be robust, the results that show the effects the submerged wire and the load have on the ship is not as clear in the ship position and orientation. Although, these effects are significant in uncontrolled states such as the roll angle and the unfiltered position and orientation measurements in addition to the corresponding rates that are fed into the wave filter. Figure 5.17 showed that the total energy consumption of the thruster system was only slightly larger in the case with crane system as compared to the case without, mostly due to the good control laws and the wave filter. However, the results show clearly the importance of running such simulations for testing control laws and filters in various scenarios. Further work may include other more high fidelity models of subsystems on-board the vessel, such as a power plant, thruster drives and a more sophisticated thrust allocation. This will be treated in the following chapters.

CHAPTER 6

Power Plant Modeling

This chapter is based on a restructured version of [39, see **P7** in section 1.6] where a marine power plant including two generators driven by two auxiliary diesel engines is derived. The main focus in this chapter is given to control of the power plant. The generators themselves are the same as those studied in chapter 4, also being hybrid causality models. In contrast to the generator model presented in the case study in Example 4.8, the marine power plant is here implemented as a bond graph model and the circuit breakers are assumed to have dynamics such that they can be both opened and closed. The overall numerical stability of the marine power plant is discussed based on the causality orientations of the generator models, and it turns out that one of the causality orientations has complex conjugated eigenvalues that require a significant reduction in local solver time-step size. However, this problem is solved by stabilizing one generator causality orientation with the other causality orientation when both generators are active. When only one generator is active it is situated with the most stable causality orientation, which is the causality required when considering the power grid as weak. Hence, this argue for the fact that if a small capacitance were added to the weak power grid, enabling static causality generator models, the simulation would be slow because the static causality generator models would have the most unstable causality orientation, in addition to the low time constant the power grid capacitance would bring forth. The presented marine power plant is solved as a continuous system and is to be further used in co-simulations in Chapter 8.

6.1 Introduction

To date, diesel electric propulsion is the most preferred solution for propulsion generation for marine vessels with a relatively large change in load conditions on a daily basis [151]. This is mostly due to its flexibility and, in general, low emissions [152], even though conversion losses may become quite significant. For a marine

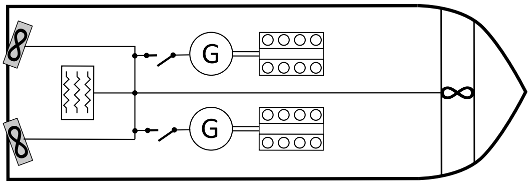


Figure 6.1: Marine vessel equipped with two gensets, two azimuth thrusters and one tunnel thruster, in addition to auxiliary loads such as hotel loads

vessel all systems that produce electrical power constitute the power plant. Different diesel engine and generator configurations, here referred to as *gensets*, can be used in a marine power plant, depending on the criteria set by the ship-owner and the classification authorities. One such configuration is shown in Figure 6.1 where two gensets constitute the marine power plant for a vessel equipped with two azimuth thrusters placed at the stern and one tunnel thruster in the bow. For reference, this thruster configuration is studied in [40], see Chapter 7, with respect to optimal thrust allocation control. Such a marine power plant is often tailored for each marine vessel and a mathematical model of the power plant is a good tool in the design process, enabling simulations of various load conditions, due to different vessel operations. One challenge when it comes to marine power plant modeling is proper control, at least if transient power plant operations are considered, e.g. a constantly changing power demand causing starting and stopping of gensets, and synchronization of gensets when being activated. A marine power plant model that facilitates such studies is the main topic presented in this chapter.

A diesel-electric marine power plant consists in general of diesel engines and electrical machineries [153], such as generators and electrical motors, which on a component basis have been studied thoroughly in the literature. In [111, 154], a two-axis bond graph model representing synchronous electrical machines is presented and studied. This model, given in the $(d, q, 0)$-reference frame, is also thoroughly analysed in [155], where equivalent circuit diagrams are also given, along with different model fidelities and model reduction techniques. Such model reductions are also studied in [156, 157]. In [155], stability and control of such systems are treated as well, and in [158] the sensitivity of eigenvalues is studied and in [159] the Nyquist stability criterion is used to assure stability in DC power systems. When it comes to overall power plant control, functionalities for synchronizing gensets and for load sharing are important. In [160], active synchronizing control of a microgrid is proposed and studied, while in [161], load sharing control is developed based on droop control and average power control.

Power grids in marine power plants are often characterized as weak, as opposed to onshore ones, which means that in practice, a large power peak in the power grid may change the rms voltage. This is because capacitive effects in the power grid itself are small and often negligible. However, often when modeling such weak power grids, a small capacitive effect is added in order to set the power grid voltage, mathematically speaking. This means that all power grid consumers and producers can be modeled with electrical current as the model output and the power grid voltage as model input. Nevertheless, such a small capacitive effect would introduce a small time constant which often stiffens the system and increases the time to solve the total system in a simulation, and is hardly wanted when trying to achieve real-time simulations.

By neglecting the capacitive effect in the power grid model itself the small time constant disappears. However, then the power grid voltage must be set by one of the power grid components. Consequently, in a generic marine power plant model where arbitrary power producers can be active, all power producers should have the possibility to set the power grid voltage, though only one at a time, if no loading model in parallel is implemented for providing the voltage. A model that has the ability to change input and output variable(s) online during a simulation is hereafter referred to as a *hybrid causality model*. Such a model switches between having *integral causality* and *differential causality*. The concept of causality is thoroughly elaborated in [25] and in Chapter 4, and will not be given any particular attention here. The reason for using hybrid causality generator models in this context is because we then always have one component in the marine power plant that can provide the voltage, and hence, no small capacitive effect introducing a small time constant is needed. However, other problems are introduced, such as solving the differential causality part of the model in a stable and fast manner without algebraic loops. This will be given more attention in section 6.2.4.

In the literature, hybrid causality models are also referred to as switched models [122] and hybrid models, in general. In [162], a theory for modeling discontinuities in models is presented, and in general, treats most kinds of hybrid dynamical models, and in [163] it is shown how simulations can be efficiently built from hybrid bond graph models. Generic synchronous generator models, having hybrid causality properties, are presented in [164] but lacks an overall numerical stability discussion, as well as a presentation of a suited power plant control structure and does not have a focus on real-time solvability.

In this chapter, a hybrid formulation of the well-known synchronous generator model in the $(d, q, 0)$-reference frame, as first presented in [164] in bond graphs, is further studied with respect to numerical stability, power management and control. This, in order to establish a generic model framework for simulating marine power plants with weak power grids suited for transient operations, while maintaining

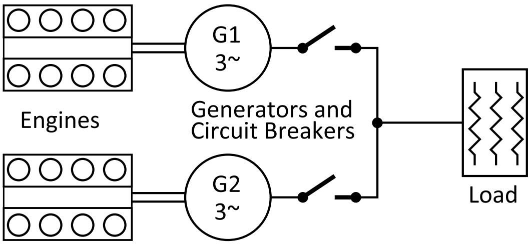


Figure 6.2: Marine power plant model

computational efficiency for real-time applications. This is significant because this topic is not overrepresented in the literature and such a generic framework gives great advantages when e.g. studying marine offshore vessels in demanding operations, as in [38], see Chapter 5, where interactions between the equipment and the power plant is important. If hardware in the simulation loop are included, such as in [41], it is also important that the models can be simulated in real-time. The proposed marine power plant framework also provides generic connections in both the $(d, q, 0)$- and the (a, b, c)-reference frame such that electrical equipment, e.g. azimuth thrusters, can be connected directly. Hence, the proposed power plant framework is a stepping stone for fast solvable total marine vessel simulators, as will be discussed in Chapter 8. A marine power plant consisting of two generic three-phased synchronous generator models, stiffly connected through a weak power grid, as shown in Figure 6.2, will be used as a case study in this chapter, and will be modeled using bond graph theory [25].

6.1.1 Outline

In the next section the hybrid causality generator model is presented in detail and analysed with respect to numerical stability when using the Euler integration method. Also, additional models such as auxiliary diesel engines and circuit breakers are presented. In section 6.3 simple control systems and strategies needed for running a marine power plant is presented. A case study of a marine power plant including two gensets are studied and simulated in section 6.4. Lastly, a conclusion is made in section 6.5.

6.2 Hybrid Generator Modeling

The hybrid generator models to be used are given in the $(d, q, 0)$-reference frame as in [164]. To keep it generic, as well as the ability to display simulation results in the (a, b, c)-reference frame, these models should have the ability to connect to the (a, b, c)-reference frame, which means that a power conservative transformation between the two reference frames is of interest.

6.2.1 Reference Frame Transformation

According to [111, 165], the $(d, q, 0)$-reference frame is related to the (a, b, c)-reference frame through the phase angle θ such that

$$\begin{aligned} \boldsymbol{u}_{d,q,0} &= A(\theta)\boldsymbol{u}_{a,b,c} \\ \boldsymbol{i}_{d,q,0} &= A^{-1}(\theta)\boldsymbol{i}_{a,b,c} \end{aligned} \tag{6.1}$$

where $\boldsymbol{u}_{i,j,k} = [u_i, u_j, u_k]^\top$, $\boldsymbol{i}_{i,j,k} = [i_i, i_j, i_k]^\top$, and

$$A(\theta) = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\theta) & \cos\left(\theta - \frac{2}{3}\pi\right) & \cos\left(\theta - \frac{4}{3}\pi\right) \\ -\sin(\theta) & -\sin\left(\theta - \frac{2}{3}\pi\right) & -\sin\left(\theta - \frac{4}{3}\pi\right) \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \tag{6.2}$$

is a power-conserving transformation matrix. The phase angle θ is defined as

$$\theta = \int_0^t f_{PG} 2\pi dt \tag{6.3}$$

Here, f_{PG} is the power grid frequency. In order for this transformation to be power conservative, it follows directly that $A(\theta)^{-1} = A(\theta)^\top$ [155]. Here, it is assumed that $u_0 = 0$. In other words, u_d, u_q and f_{PG} is a representative set of variables for u_a, u_b and u_c.

6.2.2 Generator Model with Current as Output

The dynamics of a generator with current as output can be expressed according to [164] as

$$\dot{\boldsymbol{\psi}} = -\omega_m \boldsymbol{D}\boldsymbol{\psi} - \boldsymbol{R}\boldsymbol{i} + \boldsymbol{E}\boldsymbol{u}_{\boldsymbol{d},\boldsymbol{q}} + \boldsymbol{b}u_f \tag{6.4a}$$

$$\boldsymbol{i} = \boldsymbol{L}^{-1}\boldsymbol{\psi} \tag{6.4b}$$

where ω_m is the engine speed, $\boldsymbol{\psi} = [\psi_d, \psi_q, \psi_f, \psi_D, \psi_Q]^\top$ is the magnetic fluxes for d, q, the field and the damping in d and q, respectively, $\boldsymbol{i} = [i_d, i_q, i_f, i_D, i_Q]^\top$

is the current vector, u_f is the field voltage that controls the generator and

$$\boldsymbol{D} = \begin{bmatrix} 0 & -n_p & 0 & 0 & 0 \\ n_p & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \ \boldsymbol{E} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\boldsymbol{R} = \begin{bmatrix} R_d & 0 & 0 & 0 & 0 \\ 0 & R_q & 0 & 0 & 0 \\ 0 & 0 & R_f & 0 & 0 \\ 0 & 0 & 0 & R_D & 0 \\ 0 & 0 & 0 & 0 & R_Q \end{bmatrix}, \ \boldsymbol{b} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{6.5}$$

$$\boldsymbol{L} = \begin{bmatrix} L_d & 0 & L_{df} & L_{dD} & 0 \\ 0 & L_q & 0 & 0 & L_{qQ} \\ L_{df} & 0 & L_f & L_{fD} & 0 \\ L_{dD} & 0 & L_{fD} & L_D & 0 \\ 0 & L_{qQ} & 0 & 0 & L_Q \end{bmatrix}$$

Here, $\boldsymbol{R}$ is the internal resistance matrix, $\boldsymbol{L}$ is the inductance matrix and n_p is the number of pole pairs in the generator. The electromagnetic torque feedback to the engine is given as

$$T_e = (\psi_d i_q - \psi_q i_d) n_p \tag{6.6}$$

6.2.3 Generator Model with Voltage as Output

The equations in (6.4) and (6.5) show that the differential equations are coupled, which means that the entire system of differential equations must be altered in order to obtain voltage as output in the generator model. This means that u_d and u_q should be calculated from ψ_d, ψ_q, i_f, i_D and i_Q. Thus, two of the differential equations in (6.4) are given differential causality, meaning that integration is replaced by differentiation when solving the model. By separating these two equations from the differential equations, they may be written as

$$\boldsymbol{u}_{d,q} = \dot{\boldsymbol{\psi}}_{d,q} + \omega_m \boldsymbol{D}_{d,q} \boldsymbol{\psi}_{d,q} + \boldsymbol{R}_{d,q} \boldsymbol{i}_{d,q} \tag{6.7}$$

where $\boldsymbol{u}_{d,q} = [u_d, u_q]^\top$, $\boldsymbol{\psi}_{d,q} = [\psi_d, \psi_q]^\top$, $\boldsymbol{i}_{d,q} = [i_d, i_q]^\top$ and

$$\boldsymbol{R}_{d,q} = \begin{bmatrix} R_d & 0 \\ 0 & R_q \end{bmatrix}, \ \boldsymbol{D}_{d,q} = \begin{bmatrix} 0 & -n_p \\ n_p & 0 \end{bmatrix} \tag{6.8}$$

It is then possible to rearrange (6.4b) such that

$$\begin{bmatrix} \boldsymbol{\psi}_{d,q} \\ \boldsymbol{i}_{f,D,Q} \end{bmatrix} = \boldsymbol{Z} \begin{bmatrix} \boldsymbol{i}_{d,q} \\ \boldsymbol{\psi}_{f,D,Q} \end{bmatrix} \tag{6.9}$$

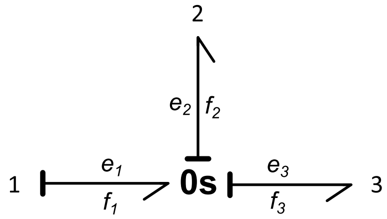


Figure 6.3: Switched 0-junction in bond graphs

which means that $\psi_{d,q}$ can be calculated from (6.9), differentiated and inserted into (6.7) to find $\boldsymbol{u}_{d,q}$. The current vector $\boldsymbol{i}_{f,D,Q}$ is found by first obtaining $\dot{\boldsymbol{\psi}}_{f,D,Q}$ from a reduced version of (6.4), integrating and inserting into (6.9). Note that $\boldsymbol{Z}$ is a rearranged version of $\boldsymbol{L}$, and can be found by solving (6.4b) with respect to $\boldsymbol{\psi}_{d,q}$ and $\boldsymbol{i}_{f,D,Q}$, which has been done in Chapter 4, see (4.44).

The electromagnetic torque given in feedback to the engine driving the generator is still as given in (6.6). Note that these two sets of equations describing the generator dynamics do not include saturation of the magnetic fluxes. This can be added by including a saturation function that saturates the magnetic fluxes after integration.

6.2.4 Hybrid Causality Model

The two different causality representations of the generator model can be put together to form the hybrid causality generator model, as done in [164] and in Chapter 4. Since causality switching happens in discrete time events, the power during switching should be conserved. For this reason, the input and the output from the old causality configuration must be inherited and used as initial conditions in the new causality configuration.

Discrete Switches

In addition to inheritance of variables, different switches are needed. The model input and output ports must be controlled by switches that routes signals from one model causality to another to make sure that the connectivity specifications in the model environment are not violated. Such switches are also needed to model the circuit breakers that connect the generators to the power grid. In [123] a design for such switches are proposed and derived using bond graph theory.

Here, only one type of switch is needed, namely a *flow switch*. This switch is often referred to as a switched 0-junction, an **0s**-junction. This name comes from its mnemonic symbolic representation in bond graph theory. The switched 0-junction is shown in Figure 6.3 in bond graphs and in Figure 6.4 in block diagrams.

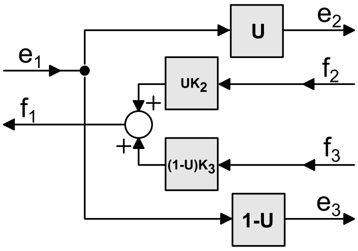


Figure 6.4: Switched 0-junction in block diagrams

The equations in the switched 0-junction are given as

$$
\begin{aligned}
e_1 =& U e_2 + (1-U) e_3 \\
f_1 =& U K_2 f_2 + (1-U) K_3 f_3
\end{aligned}
\tag{6.10}
$$

where

$$
U = \begin{cases} 1, & \text{when port 2 is active} \\ 0, & \text{when port 3 is active} \end{cases}
\tag{6.11}
$$

and

$$
K_2, K_3 = \begin{cases} 1, & \text{for addition} \\ -1, & \text{for subtraction} \end{cases}
\tag{6.12}
$$

Numerical Differentiation using Filter

In comparison to the current causality model that has five integral causalities, the voltage causality model has two differential causalities and three integral causalities. It is always preferable to have integral causality due to the use of explicit solvers, and to avoid the direct differentiation of $\psi_{d,q}$. In Chapter 4 it was proposed to differentiate $\psi_{d,q}$ by using a low-pass filter with derivative effect given as

$$
H(s) = \frac{s}{1+Ts}
\tag{6.13}
$$

where the time constant T is the bandwidth time constant of the filter. This transfer function changes the differential causalities to integral causalities when solved in the time plane, which may enable explicit fixed-step solvers such as the Euler integration method, as used here. Note that the poles and the zeros of the dynamical system also set requirements for which solver to use. As it turns out, the transfer function-based differentiation also has a positive effect on solvers due to the low-pass filtering effect, and will filter out high frequent oscillations due to numerical errors, having negligible effects on the simulation results when tuned properly. The use of this transfer function for rearranging differential causalities to integral causalities is thoroughly analysed in Chapter 4 where parallels are

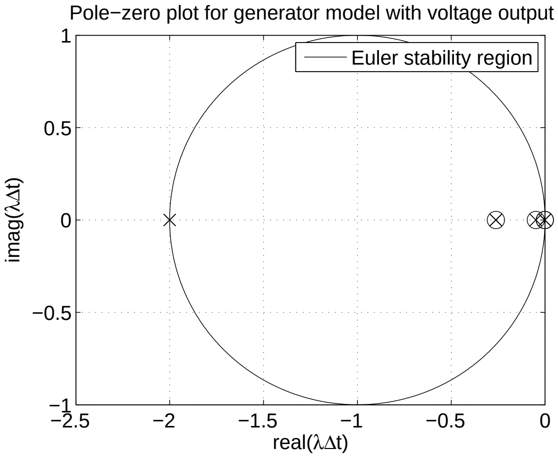


Figure 6.5: Only internal generator load ($i_d = i_q = 0$)

drawn towards adding compliance and dissipative dynamical effects for removing differential causalities.

Numerical Stability and Solver Time Step

The generator parameters used in this chapter are given in Table 4.1. By assuming a constant generator speed corresponding to $60\,\mathrm{Hz}$ in power grid frequency, corresponding to an engine speed of $720\,\mathrm{rpm}$ when having five pole pairs, open-loop pole-zero plots can be constructed for the two generator state space models given in sections 6.2.2 and 6.2.3. The pole-zero plots for the generator models with voltage output and current output are given in Figures 6.5 and 6.6, respectively.

As can be seen in Figure 6.5 the generator model with voltage outputs has the poles and the zeros inside the stability region for the Euler integration method when $i_d = i_q = 0$ and $\Delta t \leqslant 0.002\,\mathrm{s}$, where Δt is the solver time step. This indicates that the Euler integration method can be used to solve the model as long as the model environment does not introduce poles and zeros that are outside the stability region. The reasons for using the Euler integration method here are that it is fast when a reasonable step size can be used, and since it is a fixed-step solver it is easy to control in hybrid model simulations, where integrator resets and state inheritances are important. Also, when the system is proven stable with the Euler integration method, the results when using Runge-Kutta integration methods are also stable [166]. Note that implicit numerical solvers can be used as well, but is harder to control when it comes to integrator resets and state inheritances, and hence, more implementation work must be expected. The same goes for variable step-sizes where zero-crossing detection must be implemented in order to properly

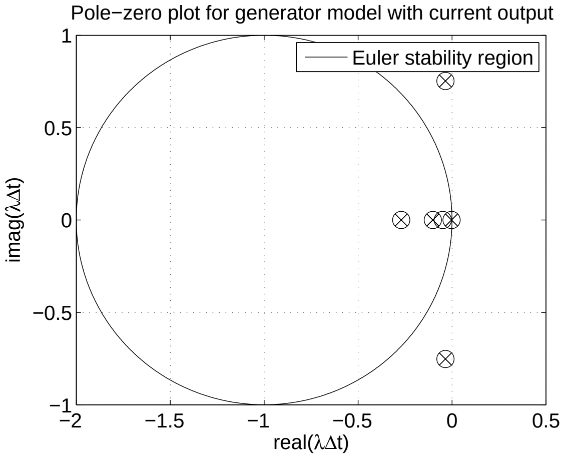


Figure 6.6: Only internal generator load ($i_d = i_q = 0$)

handle discrete events.

Figure 6.6 shows that by using the same time step, $\Delta t = 0.002\,\mathrm{s}$, the Euler integration method will fail to converge to any solution due to the poles and zeros outside the stability region. As a matter of fact, the time step must be reduced by a factor of approximately 10 in order to stabilize the zero input case, and to approximately $\Delta t = 5 \cdot 10^{-12}\,\mathrm{s}$ if an open circuit load of $R = 10^8$ is to be considered. This is without taking the model environment into consideration. If even possible, such a low time step would result in a long simulation time when solving the system, which is not desired, especially when it comes to achieving real-time simulations. Nonetheless, this problem may be solved by choosing the simulation setup or through proper pole-zero cancelling control designs, with the first mentioned treated in this chapter.

By recognizing that the complex conjugated poles and zeros are dependent on the $[i_d, u_d]$- and $[i_q, u_q]$-pairs, the model will be stable when solved by the Euler integration method if u_d and u_q are stabilized by the generator model environment. Since the poles in the generator model with voltage output are almost unaffected by large open-circuit resistances, the voltage output causality model can be used to stabilize the model with current output causality. In other words, the generator model with current as output should only be used in parallel with a generator model that has voltage output causality, because if the generator with voltage outputs is stable, the voltage inputs u_d and u_q to the generator model with current output causality will also be stable, and thus, stabilize the entire generator model. This can also be shown in Figure 6.7 and Figure 6.8.

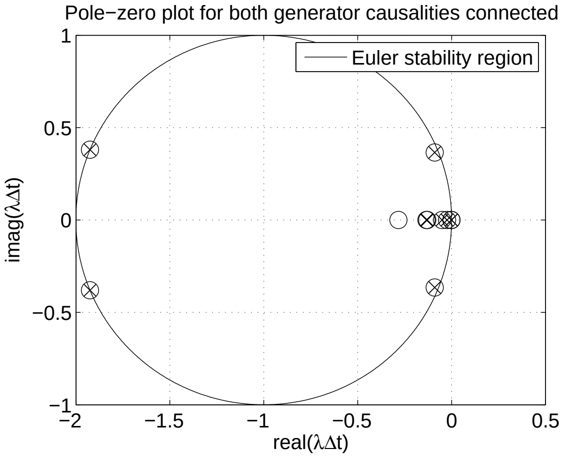


Figure 6.7: Euler stability regions for both causality models connected having only internal generator load ($i_d = i_q = 0$)

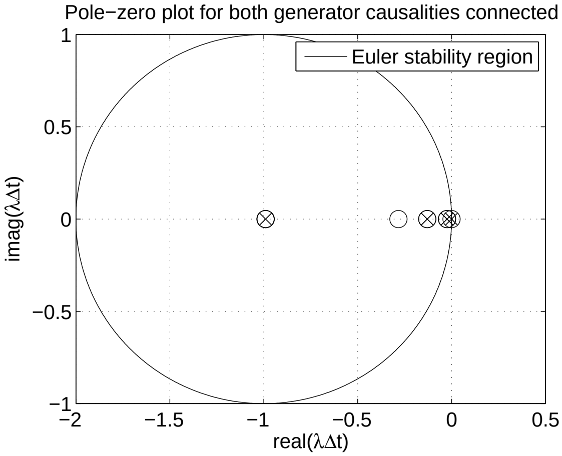


Figure 6.8: Euler stability regions for both causality models connected having open circuit load, $R = 1 \cdot 10^8 \Omega$

It can be seen in Figure 6.7 that when the time step is reduced to $\Delta t = 0.00099\,\mathrm{s}$, the two causality models connected together are stable using the Euler integration method and the parameters given in Table 4.1. The two complex conjugated poles and zeros in the figure are dependent on the time constant chosen in the transfer function given in (6.13). This means that when the time step is chosen, such that $\Delta t < T$ and the controllers and model environment are stable, the whole power plant model would be stable for all reasonable power plant loads. As a matter of fact, by increasing the power grid load more damping is added and the complex conjugated poles shown in Figure 6.7 will become real and be moved towards the origin, as shown in Figure 6.8. This means a larger time step can be used when running the generators against a power grid consumer, and indicates that running the generators in open circuit mode would be the operation that requires the smallest time steps. Hence, when ramping up a generator and synchronizing the generator, a model with voltage output should be used, and when another generator sets the grid voltages the generator model should switch causality when the circuit breaker is closed.

It is also important that the solver time step size is chosen such that it is possible to capture the true dynamic characteristics of both the generator models and the power grid load. According to [167] the proper time-step duration must be determined to accurately represent the system frequency response up to the fastest transient of interest. For example, if the power grid load is modeled in the (a, b, c)-reference frame, having a frequency of 60 Hz, the time step size must be set such that $\Delta t \leqslant \frac{1}{120}$ s in order to catch the main characteristics of the alternating currents and voltages according to the Nyquist sampling criterion [29]. In addition, the solver time step size must also be set low enough to replicate the loading characteristics themselves. However, it is believed that these characteristics will have a lower frequency than the power grid frequency, such as the meeting frequency between the wave loads and the vessel when considering a vessel in transit. Also, when studying higher order harmonic distortions in the power grid, the solver time step size must be adjusted correspondingly.

Implementation and Phase Angle

The resulting hybrid generator model is shown in a bond graph implementation in Figure 6.9. Note that $\hat{\theta}$ given in the figure is not necessary equal to θ. Since the $(d, q, 0)$-reference frame is more or less a DC representation of the actual AC system, the phase angles between the generators must be included before connecting the generators to a common power grid in order to enable phase synchronization and power sharing, which will be elaborated later on.

In general, the phase transformation between the power grid and a generator i with voltage outputs is given as

$$\boldsymbol{u}_{PG} = \boldsymbol{S}(\phi_i)\boldsymbol{u}_{Gi} \tag{6.14}$$

where $\boldsymbol{u}_{PG}$ is a vector containing the d and q power grid voltages, $\boldsymbol{u}_{Gi}$ is a vector containing the d and q generator voltages and

$$\boldsymbol{S}(\phi_i) = \begin{bmatrix} \cos(\phi_i) & -\sin(\phi_i) \\ \sin(\phi_i) & \cos(\phi_i) \end{bmatrix} \tag{6.15}$$

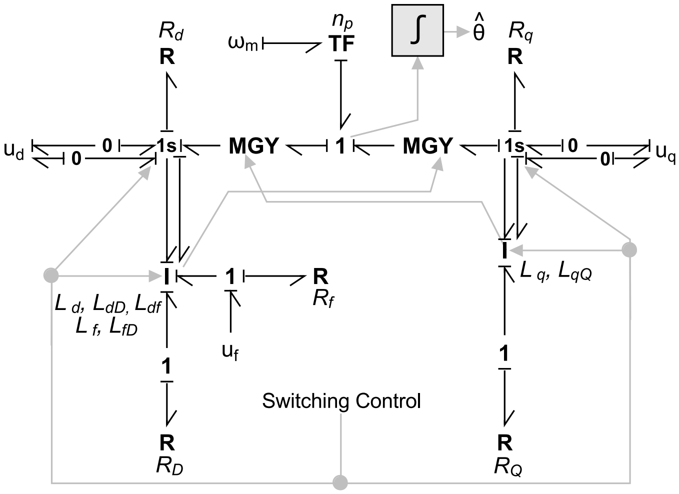


Figure 6.9: Bond graph model of hybrid generator model

where ϕ_i is the phase difference between the leading generator and generator i,

$$\phi_i = \int_0^t (\Delta\omega_{m_i})dt = \hat{\theta}_l - \hat{\theta}_i \tag{6.16}$$

Here, $\Delta\omega_{m_i} = \omega_{m,l} - \omega_{m_i}$ is the difference in motor velocity between the two engine speeds driving the generators and $\omega_{m,l}$ and $\hat{\theta}_l$ are the speed and phase angle of the leading generator, respectively. This difference in phase angles is usually only calculated when generators are active, meaning that they are either preparing for synchronization or already connected to the power grid, which also means that an integrator reset algorithm should be implemented.

The phase angle used in the transformation between the (a, b, c)- and $(d, q, 0)$ reference frame, given in (6.2), must then account for ϕ. Hence, the phase angle for a generator i is then given as

$$\theta_{Gi} = 2\pi \int_0^t f_{Gi}dt + \phi_i \tag{6.17}$$

where f_{Gi} is the frequency for generator i and where ϕ_i is as defined in (6.16). The different transformations between the power grid and the two causality models are summarized in Table 6.1.

6.2.5 Additional Models needed for Simulation

To be able to set up a simulation testing for the generators, additional models must be included, e.g. circuit breakers, power grids and engine models, in addition to controllers and a power management system.

Table 6.1: Phase transformations for generator i.

	Voltage Output	**Current Output**
Voltage	$\boldsymbol{u}_{PG} = \boldsymbol{S}(\phi_i)\boldsymbol{u}_{Gi}$	$\boldsymbol{u}_{Gi} = \boldsymbol{S}^{-1}(\phi_i)\boldsymbol{u}_{PG}$
Current	$\boldsymbol{i}_{Gi} = \boldsymbol{S}^{-1}(\phi_i)\boldsymbol{i}_{PG}$	$\boldsymbol{i}_{PG} = \boldsymbol{S}(\phi_i)\boldsymbol{i}_{Gi}$

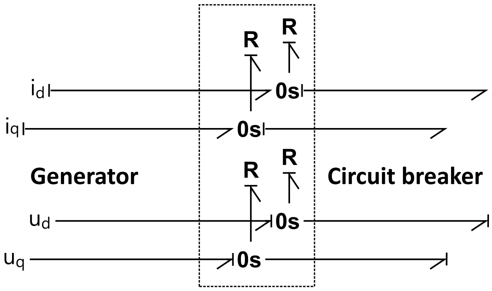


Figure 6.10: Bond graph model of circuit breaker

Circuit Breakers and Power Grid Model

Multiple switched 0-junctions are used to model the circuit breakers and the power grid. In the circuit breaker, model port 1 of the switched 0-junction in Figure 6.4 is connected to the generator model output, port 2 is connected to an open circuit load and port 3 is connected to the power grid. Note that several switched 0-junctions are needed in one circuit breaker model since the generator model has a total of four inputs and outputs, due to the two causality configurations connected to the grid. These circuit breakers are connected to each other and to the consumer models in the power grid through another set of switched 0-junctions, due to the causality switching in the generator models. The total model of a circuit breaker is shown in Figure 6.10 in bond graphs. Note that each switched 0-junction is also given a control signal, U, as in (6.10), and that additional switched 0-junctions are needed to connect several gensets to a common power grid.

Simplified Engine Model

The auxiliary engine models used to drive the generator models are based on simple equations given in [168]. The effective engine power is given as

$$P_e = \dot{m}_f h_n \eta = T_m \omega_m \tag{6.18}$$

where $\dot{m}_f$ is the fuel flow rate, h_n is the lower heating value of the fuel, η is the effective thermal efficiency, T_m is the torque and ω_m is the engine speed. By rearranging the equation, the mean torque generated by the combustion process can be expressed as

$$T_m = \frac{\dot{m}_f h_n \eta}{\omega_m} \tag{6.19}$$

The fuel flow rate may be expressed as

$$\dot{m}_f = m_{inj} \frac{\omega_m}{2\pi k} \tag{6.20}$$

where m_{inj} is the amount of fuel injected per cycle and k is a parameter distinguishing two-stroke engines from four-stroke engines, $k = 1$ for two-stroke and $k = 2$ for four-stroke. The efficiency can be expressed as

$$\eta = \frac{1}{b_e(P_e) h_n} \tag{6.21}$$

where $b_e(P_e)$ is the brake specific fuel consumption as a function of effective engine power, and can be measured for a specific engine given the engine speed. By assuming a four-stroke engine, the torque can then be expressed as

$$T_m = \frac{m_{inj}}{4\pi b_e(P_e)} \tag{6.22}$$

The set of differential equations representing the auxiliary engine model is then to be given as

$$\dot{\theta}_m = \omega_m \tag{6.23a}$$

$$\dot{\omega}_m = \frac{1}{J_m + J_G} \left(T_m - b_f \omega_m - b_b \omega_m^n - T_e \right) \tag{6.23b}$$

where θ_m is the engine angle, J_m is the inertia of the engine, J_G is the inertia of the generator, T_e is the electromagnetic torque as given in (6.6), b_f is a friction parameter, b_b is the braking effect when the engine is choked, which happens when no fuel is injected into the engine due to the pumping work and n is a small number, typically ~ 0.1. Note that $b_b := 0$ when $m_{inj} \neq 0$. To make the transition smooth, the value of b_b is low-pass filtered, having a small time constant before being used in (6.23).

Power Grid Consumer

The power grid consumer has the role of loading the power grid. This can be a set of thrusters, as well as the auxiliary systems needed to power the ship. Here, it is assumed that the active- and reactive power consumption can be set and based on the power grid voltage, a current can be given in feedback. The active power and reactive power in the $[d, q, 0]$-reference frame are given as

$$P = \boldsymbol{u}_{d,q}^T \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{i}_{d,q} \tag{6.24a}$$

$$Q = \boldsymbol{u}_{d,q}^T \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \boldsymbol{i}_{d,q}, \tag{6.24b}$$

respectively. Note that reactive power is related to impedance in the (a, b, c)-reference frame. By solving (6.24a) and (6.24b) with respect to $\boldsymbol{i}_{d,q}$ the current given in feedback is given as

$$\boldsymbol{i}_{d,q} = \frac{1}{||\boldsymbol{u}_{d,q}||_2^2} \begin{bmatrix} Pu_d + Qu_q \\ Pu_q - Qu_d \end{bmatrix} \tag{6.25}$$

where

$$||\boldsymbol{u}_{d,q}||_2^2 = u_d^2 + u_q^2 \tag{6.26}$$

is the square of the $\mathcal{L}_2$-norm. To avoid dividing by zero at the start of the simulation, a small number is added to the denominator. Also, two low-pass filters are used to filter the input voltages in order to avoid algebraic loops.

Power Management System, PMS

Power management systems are in reality quite complex. In this chapter, the PMS is treated as a collection of controllers with some automated decision-making functionalities. This will be highlighted in section 6.3.3.

6.3 Power Plant Control

In general, systems that produce electric power require proper control dependent on the operation in order to run satisfactory [169]. For generators, proper control of the generator phase synchronization, voltage control and load sharing, or power sharing, are crucial for running the power plant within its limits.

In general, a genset only has two controllable variables, namely the engine speed and the field voltage, which means that more than one objective must be controlled through each controllable variable. In single model systems this is not preferred since it often sacrifices stability or degradation of the control strategies. However, in power plants, a single genset connected to a power grid only has to control the frequency and rms voltage, which can be separately done through the two controllable variables. It is only when two or more gensets are connected that multiple control strategies must be merged into the two control variables. This can be done in a stable manner in such systems due to the stiff coupling between gensets, since the difference in frequency, rms voltage and phase can be indirectly controlled.

In this section, different controllers needed to run the model shown in Figure 6.2 are presented.

6.3.1 Automatic Voltage Regulator and Reactive Power Sharing

Both the rms voltage and reactive power sharing are controlled through the field voltage, but the reactive power sharing control is only activated when two or more generators are synchronized and connected together.

Automatic Voltage Regulator (AVR)

The voltage magnitude, or the rms voltage, is regulated in each generator by an automatic voltage regulator (AVR). The AVR regulates the voltage magnitude through the generator field voltage, which sets the magnetic field in the stator. The rms voltage in the $(d, q, 0)$-reference frame for a generator i is given as

$$V_{rms_i} = \sqrt{\frac{2}{3}} ||\boldsymbol{u}_{dq,i}||_2 \tag{6.27}$$

Note that $\boldsymbol{u}_{dq,i}$ is the voltage vector for the active causality model of the generator. There exists many different control laws for AVRs, such as the one presented in [170] with pole assignment self-tuning regulator. Nevertheless, here the rms voltage is controlled by a simple PI-controller, and the controller error for generator i is defined as

$$e_{Vi} = V_{ref} - V_{rms,i} \tag{6.28}$$

where V_{ref} is the reference rms voltage. Hence, the corresponding controller output u_{Vi} can be expressed as

$$u_{Vi} = K_p^{AVR} e_{Vi} + \frac{K_p^{AVR}}{T_i^{AVR}} \int_0^t e_{Vi} dt \tag{6.29}$$

where K_p^{AVR} and T_i^{AVR} are the proportional gain and the integral time constant, respectively. In order to avoid commanding an unrealistic field voltage the controller output is saturated before being sent to the generator model. Hence, an integration anti-windup algorithm [171] is implemented in the controller as well.

Reactive Power Sharing

Control of the reactive power is found in the PMS since reactive power sharing control needs measurements from all active generators, which is why it cannot be implemented as local controllers inside each generator. Another reason is that it is only active when two or more generators are active. This means that global power plant surveillance is needed, which is a PMS functionality. Power sharing in general is thoroughly discussed in [172], and only a simple method for controlling the reactive power sharing is presented here.

By assuming that N number of generators are connected to the power grid, the reactive power sharing can be controlled by means of PI-controllers where the error for generator i is defined as

$$e_{Qi} = S_{Qi} Q_{tot} - Q_{Gi} \tag{6.30}$$

and where $S_{Qi} \in [0,1]$ is the sharing factor, Q_{Gi} is the reactive power for generator i and

$$Q_{tot} = \sum_{i=1}^{N} Q_{Gi} \tag{6.31}$$

is the total reactive power in the power grid, or the reactive power for the power grid load. Hence, the corresponding controller output u_{Qi} can be expressed as

$$u_{Qi} = K_p^Q e_{Qi} + \frac{K_p^Q}{T_i^Q} \int_0^t e_{Qi} dt \tag{6.32}$$

where K_p^Q and T_i^Q are the proportional gain and the integral time constant, respectively. Note that if equal reactive power sharing is considered, then $S_{Qi} = \frac{1}{N}$. Thus, the total field voltage that is fed to generator i is then given as

$$u_{fi} = u_{Vi} + u_{Qi} \tag{6.33}$$

Here, only two generators are considered in the power plant, and the controller errors for the reactive power sharing control for the two generators are defined as

$$e_{Q1} = S_Q Q_{tot} - Q_{G1} \tag{6.34a}$$
$$e_{Q2} = (1 - S_Q) Q_{tot} - Q_{G2} \tag{6.34b}$$

respectively, where $S_Q \in [0,1]$ is the sharing factor.

Since the reactive power sharing controllers are only activated when two or more generators are synchronized and connected to the power grid, integrator reset algorithms must also be implemented in the reactive load sharing controllers. The structure of the total rms voltage control system and the reactive power sharing control system is shown in Figure 6.11 for two active genesets.

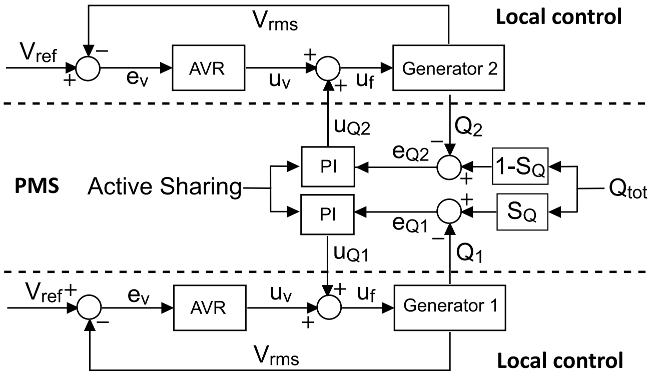


Figure 6.11: AVR and reactive power sharing, RPS, control structure for two active gensets

6.3.2 Engine Control, Active Power Sharing and Generator Synchronization

The last control variable is the engine speed driving the generator. In general, the auxiliary engine is to maintain a constant speed that corresponds to the wanted power grid frequency, but also generator synchronization and active power sharing can be done through the engine speed, the last one through e.g. *droop control* [161] or *isochronous control* [173].

Engine Control

The engine speed controller, commonly known as the *engine governor* , is in general complex [174], but is assumed here to be a simple PI-controller that controls the engine speed, such that the power grid frequency is kept more or less constant. The reference engine speed is given as

$$\omega_{ref} = \frac{2\pi f_{PG}}{n_p} \tag{6.35}$$

In comparison to the field voltage control and the reactive power control, active power sharing control and phase synchronization control are applied to the reference signal before being fed into the PI-controller. The reference fed to the controller is denoted $\omega_{ref,ni}$ for generator i, as will be defined shortly, and the controller error can be expressed as

$$e_{\omega_i} = \omega_{ref,ni} - \omega_{mi} \tag{6.36}$$

where ω_{mi} is the measured engine speed for genset i. Hence the corresponding control output $m_{inj,i}$ can be expressed as

$$m_{inj,i} = K_p^\omega e_{\omega_i} + \frac{K_p^\omega}{T_i^\omega} \int_0^t e_{\omega_i} dt \tag{6.37}$$

where K_p^ω and K_i^ω are the proportional gain and the integral gain, respectively. Also this control output is saturated in order to give realistic fuel injections to the engine. Hence, an integrator anti-windup algorithm is implemented in the controller as well.

Synchronization Control

In the (a, b, c)-reference frame, the phase synchronization controller makes the phase angles converge through controlling the engine speed of the generator that is to be synchronized. However, in the $(d, q, 0)$-reference frame, it is enough to make the d voltages converge, since the q voltages will then converge due to the AVR, in order to make the phases converge. This can be shown by setting the criterion

$$u_{al} = u_{ai} \tag{6.38}$$

where u_{al} and u_{ai} are the a voltages in the (a, b, c)-reference frame for the lead generator and the generator i, which is to be synchronized, respectively. This gives the criterion in the $(d, q, 0)$-reference frame

$$(\cos\theta_i - \cos\theta_l)u_d = (\sin\theta_i - \sin\theta_l)u_q \tag{6.39}$$

when the d voltages converge. The only solution of (6.39) for all values of u_d and u_q is $\theta_l = \theta_i$, which means that by making the d voltages converge a solution for $\theta_l = \theta_i$ exists under the restriction given in (6.39). By also requiring that $u_{bl} = u_{bi}$ and $u_{cl} = u_{ci}$ the only solution left is $\theta_l = \theta_i$ if all the criteria are to be held. This means that by making the difference in the d-voltages converge, the difference in the phase angles will also converge.

The main reason for comparing the voltages instead of the phase angles is because the causality switching requires equal voltages and currents right before and after switching, and due to the risk of numerical deviations when simulating, it is better to compare the voltages directly in the phase synchronization controller. When the phase synchronization controller is active, the control errors for phase synchronization between the lead generator l and generator i can then be given as

$$e_{PSi} = u_{dl} - u_{di}, \tag{6.40}$$

for the generator i that is to be connected to the grid. Note that the lead generator is defined as one generator that is already connected to the grid. Hence the corresponding control output u_{PSi} can be expressed as

$$\begin{aligned} u_d &= \frac{N_d^{PS}}{T_d^{PS}} \int_0^t u_{dPSi} dt \\ u_{dPSi} &= K_p^{PS} N_d^{PS} e_{PSi} - u_d \\ u_{PSi} &= K_p^{PS} e_{PSi} + u_{dPSi} \end{aligned} \tag{6.41}$$

where K_p^{PS} and N_d^{PS} and T_d^{PS} are the proportional gain, the derivative gain limitation and the derivative time constant, respectively. Note that u_d and u_{dPSi} are initially set to zero and that the control equations given in (6.41) should be calculated in the given order. Also in this controller the output is saturated in order not to give too large of deviations in the engine speeds. Hence, an integrator anti-windup algorithm is implemented as well.

Active Power Sharing

The active power grid load can be shared between multiple active gensets using either a droop control strategy or an isochronous control strategy, and the latter strategy will be considered here. The droop control strategy is a passive load sharing strategy where the generator frequency, hence the engine velocity, is reduced proportionally with an increasing active power load, typically about 3-5 $\%$ of the static power grid frequency from no-load to fully loaded. Hence, the static power grid frequency is also reduced with an increasing power grid load if not being compensated for by e.g. adaptively adjusting the no-load frequency.

In comparison to the droop control strategy the isochronous control strategy is an active load sharing strategy where all active gensets try to keep the power grid frequency constant, except when changing the active load distribution. This control strategy requires that each active genset has information about how much active power it should produce. When the active power grid load is being shared the active load sharing controllers change the engine reference signals slightly, and when gensets get different frequencies the distribution of power grid load is changed. A higher frequency gives a genset more active power, and a lower frequency reduces the active power. This means that the amount of power a given generator produces can be adjusted by subtracting an actual power measure from the engine reference speed and adding a reference power measure to the engine reference speed, such that the new engine reference speed for engine i becomes

$$\omega_{ref,ni} = \omega_{ref} + K_{aps}(P_{ref,i} - P_{mi}) \tag{6.42}$$

where ω_{ref} is the commanded engine reference speed, K_{aps} is a proportional gain, P_{mi} is the measured power and $P_{ref,i}$ is the reference power. Then, if the power is too low, the engine speed will increase such that more load is added to the respective generator. If the power is too high, the opposite occurs.

One can choose to share the load using the load magnitude or using the loading percentage for each genset. Here, the loading percentage is used. Note that equal power sharing in this context means that each genset has the same loading percentage, though not necessarily an equal load. In order for this to be true, all gensets must have equal capacity.

The engine power fraction used in active power sharing control is defined for engine i as

$$L_{mi} = \frac{T_{mi}\omega_{mi}}{P_{ei,max}} \tag{6.43}$$

where T_{mi} is the engine torque as defined in (6.22), ω_{mi} is the engine angular rate and $P_{ei,max}$ is the lowest maximally rated effective power for the engine and the generator. For a power plant with an N number of gensets, the total load fraction is defined as

$$L_{tot} = \sum_{i=1}^{N} L_{mi} \tag{6.44}$$

When active power sharing control is active, the reference power fraction is added to the reference speed and is for generator i given as

$$\omega_{drp1}(i) = K_D S_{Pi} L_{tot} \omega_{ref} \tag{6.45}$$

where K_D is a proportional gain and $S_{Pi} \in [0, 1]$ is the active power-sharing constant. Note that (6.45) is run through a low-pass filter with filter time constant T_{LP} in order to smooth the signal before being added to ω_{ref}. Also note that $S_{Pi} L_{tot} \leqslant 1$.

Next, $\omega_{drp2}(i)$ is subtracted from $\omega_{ref} + \omega_{drp1}(i)$ when active power sharing control is active, and is given as

$$\omega_{drp2}(i) = K_D L_{mi} \omega_{ref} \tag{6.46}$$

where L_{mi} is the local power fraction defined in (6.43). $\omega_{drp2}(i)$ is also low-pass filtered in order to smooth the signal. When comparing the expressions for ω_{drp1} and ω_{drp2} to (6.42), it can be seen that $K_{aps} = K_D \omega_{ref}$, $P_{mi} = L_{mi}$ and $P_{ref,i} = S_{Pi} L_{tot}$. By including the phase synchronization control the PI-controller error for the engine driving generator i is then rewritten as

$$e_{\omega_i} = \omega_{ref} + \omega_{drp1}(i) - \omega_{drp2}(i) - u_{PSi} - \omega_{mi} \tag{6.47}$$

where u_{PSi} is as defined in section 6.3.2. The PI control law is as given in (6.37).

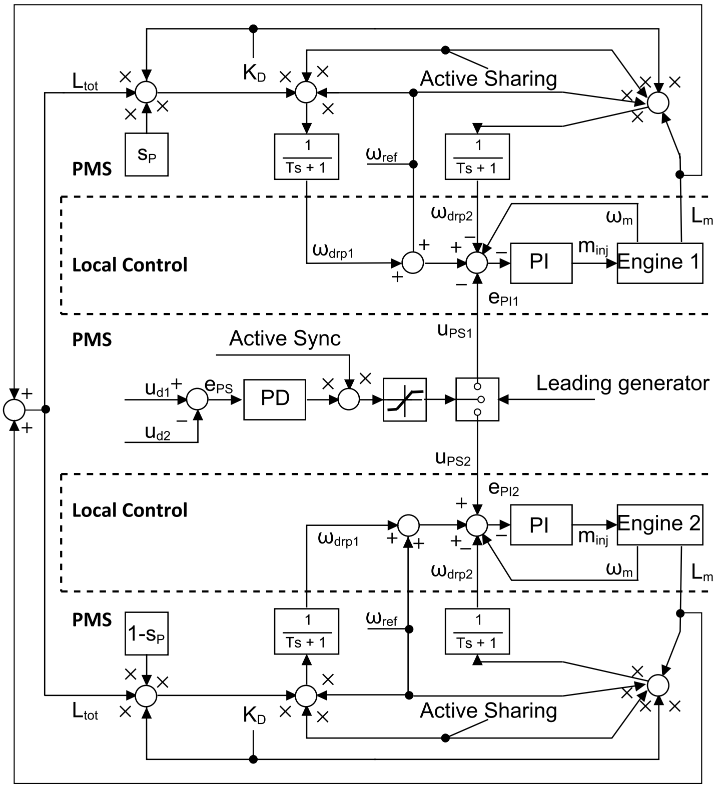


Figure 6.12: Engine speed control, active power sharing and phase synchronization for two gensets. Note that the leading generator, the one active having voltage output, is not affected by synchronization control

Figure 6.12 shows a graphical representation of the engine speed control, active power sharing control and phase synchronization for genset 1 and genset 2. More advanced load sharing control strategies than the one presented here can also be applied, e.g. as the one presented in [161] where droop control and average power control are combined.

6.3.3 Causality- and Simulation Control

When using hybrid models, it is important to have a *simulation controller*. In general, a simulation controller is not a controller using common sense, but rather a set of logic functions that controls the switching of the hybrid models. In this study, such a simulation controller would typically open and close circuit breakers, switch causality in the hybrid generator models and the resulting causality in the power grid. Functionality that restricts the input signals is also a common practice to implement, such as the generator setting the power grid voltage, which cannot be disconnected before model switching is performed.

The logic functions in the simulation controller are closely coupled to global controllers such as the phase synchronization controller and power sharing controller. The logics activate and deactivate these controllers, thereby ensuring that the integrators are reset in addition to evaluating the controller performances, e.g. evaluating when the phase difference between the generators is small enough to close the circuit breaker. These functions are related to the functionalities found in real power management systems and in general it is up to the modeller or the PMS vendor to decide which functionality to implement, and is usually kept secret. In this chapter, the PMS includes the functionalities needed to control the hybrid generator models, the circuit breakers and the power grid causality, and is functionally related to the phase synchronization control and power sharing control.

In particular, the PMS decides when to close the circuit when two generators are being synchronized. Here, a set of rules for closing the circuit when two generators are being synchronized, mainly based on minimizing numerical distortions when closing the circuit breakers, are implemented and given as

$$\begin{aligned} |\phi| &\leqslant \phi_{max} \\ |\frac{d\phi}{dt}| &\leqslant \dot{\phi}_{max} \\ |V_{rms,1} - V_{rms,2}| &\leqslant dV_{rms,max} \\ |f_{G,1} - f_{G,2}| &\leqslant df_{G,max} \end{aligned} \tag{6.48}$$

where ϕ_{max} is the maximal allowed phase difference, $\dot{\phi}_{max}$ is the maximal allowed derivative of the phase difference, $dV_{rms,max}$ is the maximal allowed difference in rms voltage between the two generators and $df_{G,max}$ is the maximal allowed difference in frequency between the two generators.

6.3.4 Overview of Control Structure

The control systems presented in this section are independent of the causality orientation of the generator model. This is shown in Figure 6.13 which shows a more coarser presentation of the control system structure and the connections between the PMS and the genset. The figure also shows that only the AVR and the engine speed controller, the governor, are always active, when neglecting the causality controller, while synchronization, active- and reactive load sharing are activated by the PMS when more than one generator is considered. Also note that the synchronization controller and the active load sharing adds to the reference signal for

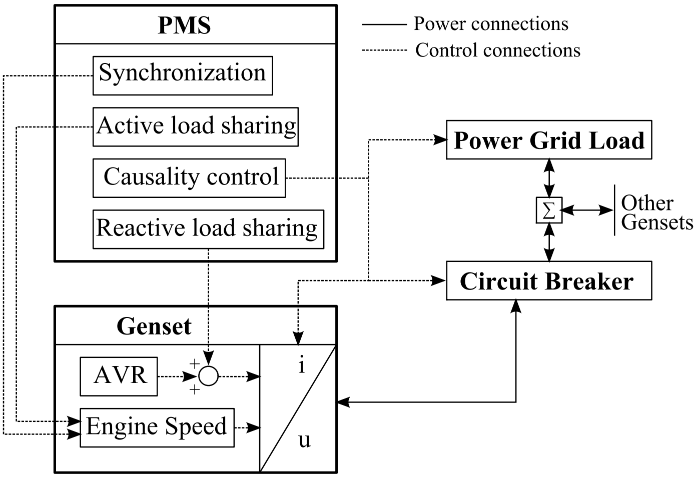


Figure 6.13: Control structure and connection between PMS and genset. Note that measurement signals needed by the PMS have been left out in the figure

the engine speed controller, while the reactive load sharing adds to the output of the AVR. Note that the block i/u in the figure denotes the two causality models of the generator including the auxiliary engine.

6.4 Simulation

To test the hybrid generator model and the corresponding control systems, a simulation of the power plant given in Figure 6.2 is to be performed. Before presenting the simulation scenario and the corresponding simulation results, the parameters to be used in the simulation are presented.

6.4.1 Hybrid Genset Models and Power Grid Load

The generators are modeled as specified in section 6.2 and are assumed to be of equal size in this case study, having a rated power output of 2438 kW with $V_{rms} = 690\,\text{V}$ and $I_{rms} = 2010\,\text{A}$. The specific generator parameters are given in Table 4.1 and the time constant in the transfer function given in (6.13) is set to 0.001 s.

The brake specific fuel consumption (BSFC) $b_e(P_e)$ for the auxiliary engines is given as a polynomial,

$$b_e(P_e) = p_0 \left(\frac{P_e}{P_{e,max}}\right)^4 + p_1 \left(\frac{P_e}{P_{e,max}}\right)^3 + p_2 \left(\frac{P_e}{P_{e,max}}\right)^2 + p_3 \frac{P_e}{P_{e,max}} + p_4 \tag{6.49}$$

Table 6.2 list the parameters for the auxiliary engines used in the simulation. Note that (6.49) is typically based on data provided from engine vendors and hence, the

Table 6.2: Main parameters regarding the auxiliary engine models

Parameter	Value
$P_{e,max}$	2010 kW
J_m	750 kgm^2
J_G	750 kgm^2
b_f	20 $^{kgm^2}/_{s}$
n	0.1 –
b_b	$\begin{cases} 0\,, & \text{if}\, m_f > 0 \\ 200\,, & \text{else} \end{cases}$
p_0	364.452 g/kWh
p_1	-1118.50 g/kWh
p_2	1417.60 g/kWh
p_3	-813.84 g/kWh
p_4	379.13 g/kWh

Table 6.3: Power grid load parameters and open circuit load parameter used in the simulation

Parameter	Value
Active power load	1 MW
Reactive power load	1 MVAr
Low-pass filter time constant	0.001 s
Open circuit load	100 MΩ

curve used here is only valid for engine loads larger than 10 % of the maximal engine loading. Also note that the fuel consumption related to auxiliary engine systems needed for running the engine is included in this BSFC curve.

The main parameters for the power grid load and the circuit breakers used in the simulation are given in Table 6.3. In addition to the constant active- and reactive power loads, biases and noise are added in order to make the simulation more realistic and to test the robustness of the total control system.

6.4.2 Control Systems

The control systems are implemented as specified in section 6.3 and the parameters are given in Table 6.4. Note that the control structure and the connections between the PMS and the gensets are as illustrated in Figure 6.13.

Table 6.4: Control parameters used in the simulation

Controller	Parameter	Value
AVR	V_{ref}	690 V
	K_p^{AVR}	5.0 -
	T_i^{AVR}	5.0 s
	Staturation	±100.0 V
Reactive	S_{Qi}	0.5 - initially
Power	K_p^Q	1.0e-5 V/VAr
Sharing	T_i^Q	1.0 s
Engine	ω_{ref} (idle)	20 π rad/s
Control	ω_{ref} (active)	24π rad/s
	K_p^ω	0.1 kgs/rad
	T_i^ω	0.1 s
	Saturation	$[0, 0.26]$ kg
Synchronization	K_p^{PS}	0.01 rad/Vs
Control	N_d^{PS}	10.0 -
	T_d^{PS}	0.008 s
Active	K_d	0.001 -
Power	S_{Pi}	0.5 - initially
Sharing	T_{LP}	0.01 s
PMS	ϕ_{max}	0.01 rad
	$\dot{\phi}_{max}$	0.1 rad/s
	$dV_{rms,max}$	0.1 V
	$df_{G,max}$	$\frac{0.025}{\pi}$ Hz

6.4.3 Simulation Scenarios

The simulation to be performed is designed to test the different control systems outlined in section 6.3 as well as the hybrid generator model and its real-time characteristics.

In the simulation the two auxiliary engines driving the generators are started in the beginning of the simulation and given the idle speed reference. At $t = 20\,\mathrm{s}$, genset

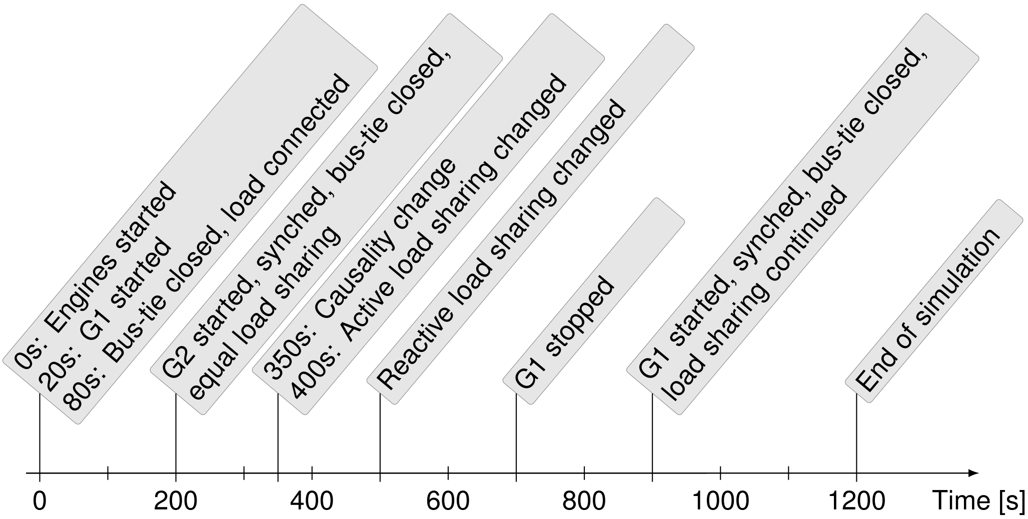

0s: Engines started
20s: G1 started
80s: Bus-tie closed, load connected
G2 started, synched, bus-tie closed,
equal load sharing
350s: Causality change
400s: Active load sharing changed
Reactive load sharing changed
G1 stopped
G1 started, synched, bus-tie closed,
load sharing continued
End of simulation
0
200
400
600
800
1000
1200
Time [s]

Figure 6.14: Simulation events. Note that G1 and G2 in the figure are abbreviations for Genset 1 and Genset 2, respectively

1 is initiated, the engine gets the nominal speed reference and the AVR is fed the rms voltage reference. The circuit breaker that connects genset 1 to the power grid is closed at $t = 80\,\mathrm{s}$. Genset 2 is initiated at $t = 200\,\mathrm{s}$, in the same manner as genset 1, and the phase synchronization is activated. When the phase difference between the gensets is within the tolerance, the circuit breaker that connects genset 2 to the power grid is closed, and the model causality is changed from voltage to current. At $t = 350\,\mathrm{s}$, both gensets change causality; while genset 2 sets the power grid voltage. The active- and reactive power sharing are then changed at $t = 400\,\mathrm{s}$ and $t = 500\,\mathrm{s}$, respectively. In the end, genset 1 is stopped at $t = 700\,\mathrm{s}$ and started again at $t = 900\,\mathrm{s}$, and the active- and reactive load sharing are initiated with the last known sharing parameters, S_P and S_Q. A summary of all the simulation events is shown in Figure 6.14.

The Euler integration method is chosen as the solver with a solver time step $\Delta t = 0.0001\,\mathrm{s}$. The time step is chosen smaller than what was stated earlier; this due to an increase in the quality of the alternating voltages and currents results.

6.4.4 Simulation Results

The rms voltages calculated from (6.27) for the two generators are shown in Figure 6.15. The first plot shows the two generator rms voltages compared to each other. As can be seen in the plot, the voltages both converge to 690 V, when the gensets are active. As illustrated in Figure 6.14, genset 1 is first started and connected to

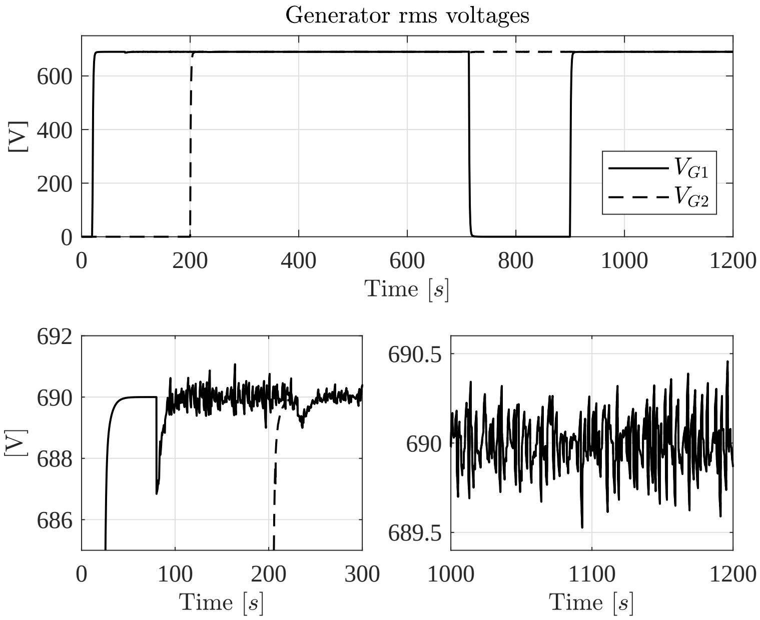


Figure 6.15: Generator rms voltages

the power grid load before genset 2 is started and synchronized to the power grid at $t = 200$ s. At $t = 350$ s, the causality configurations for the two gensets are altered before genset 1 is commanded to disconnect from the power grid at t =700 s. In the end, genset 1 is reconnected to the power grid at $t = 900$ s. The plot in the lower left-hand corner in Figure 6.15 shows a closer view of what happens with the rms voltage for generator 1, which has output voltage causality when the load and the second generator are connected. When the power grid load is connected, the rms voltage for generator 1 gets a small dip before being controlled back to its reference. The dip has a magnitude of approximately 3 V which results in a change in rms voltage of approximately $0.43\,\%$, which is well within any typical realistic tolerance, typically $\sim\, 1\,\%$ of the reference rms voltage. A small dip in the rms voltage can also be seen when the second generator is connected, which is expected to be due to the stabilization of the two connected generators, and due to the start of the load-sharing procedure. However, this dip is smaller, having a magnitude of about 1 V, approximately $0.145\,\%$ of the rms reference voltage. The last plot in the lower right-hand corner shows the rms voltage fluctuations in the time range $t \in [1000\,\text{s}, 1200\,\text{s}]$. As can be seen, the rms voltages for the two generators overlap, although some noise is present. This noise is due to the noise added in the power grid load in order to test the robustness of the power plant. The noise magnitude is quite low, at the most approximately $0.07\,\%$ of the rms voltage

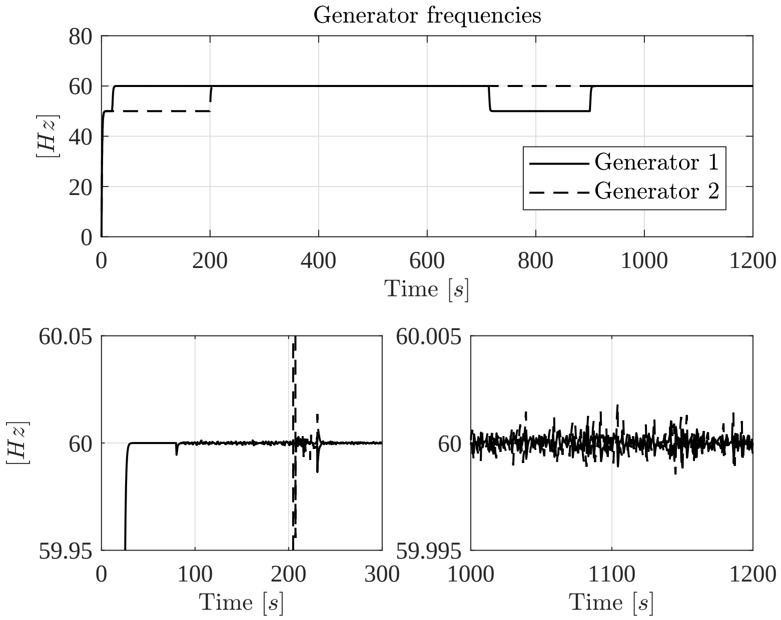


Figure 6.16: Generator frequencies

reference, but large enough to indicate stability and robustness in the total power plant model.

Figure 6.16 shows the two generator frequencies compared to each other. The first plot shows the comparison of the two generator frequencies during the entire simulation, and the characteristics are comparable to the rms voltage characteristics in the first plot in Figure 6.15, except that when a generator is disconnected from the power grid, it runs in an open circuit at an idling speed of 600 rpm. The plot given in the lower left-hand corner shows a closer view of the frequencies at the start of the simulation. The characteristic dip when the load is connected is also present here, having a magnitude of approximately 0.0055 Hz. When the second genset is started, the frequency of generator 2 overshoots the first generator frequency, due to phase synchronization, with a magnitude of approximately 0.08 Hz before converging to the desired frequency. When the two generators are connected the active power sharing characteristics can be seen at approximately $t = 230$ s. Genset 2 increases its frequency, whereas genset 1 slows down due to active power sharing control. Nonetheless, these frequency peaks are quite small, approximately 0.015 Hz in magnitude. The last plot, given in the lower right-hand corner of the figure, shows the frequency fluctuations due to the noise. Note that the two generator frequencies do not overlap as the rms voltages do in Figure 6.15,

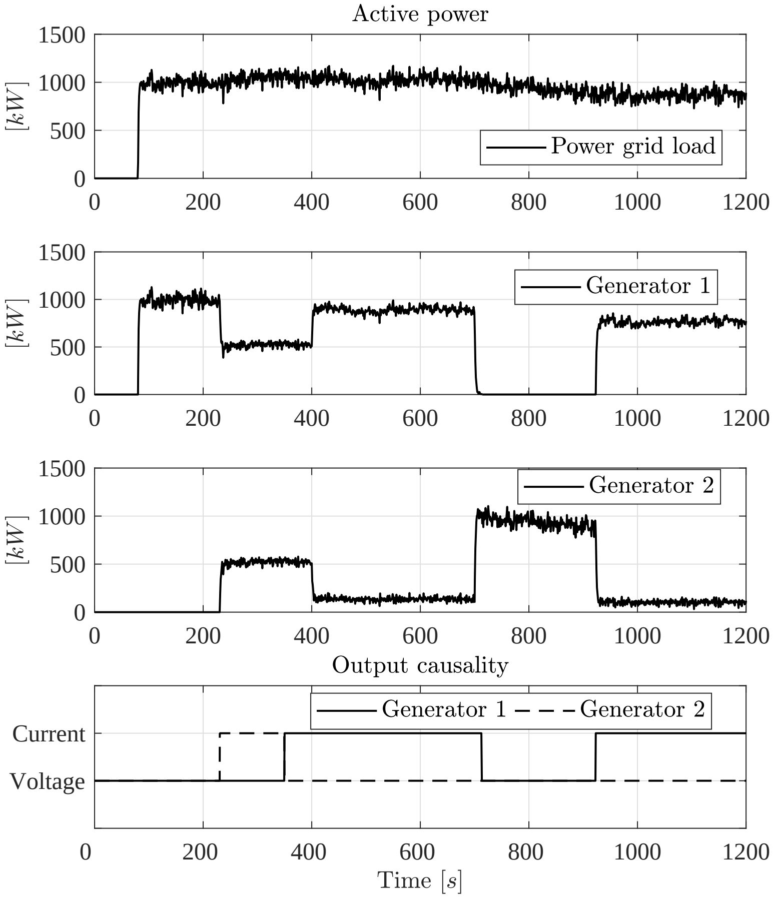


Figure 6.17: Active Power

which is expected due to the noise and the active power sharing control.

The active power production and consumption are shown in Figure 6.17, in addition to the genset output causalities. The first plot in the figure shows the active power grid load, which oscillates at around 1 MW with a relatively high frequent noise, having a maximal frequency of 1.0 Hz in addition to a slowly varying oscillating bias, modeled as the integral of white noise. The plot also shows that the load is increased relatively quickly; when connected, it takes approximately 5 s from when the load is connected until it reaches its full potential of power consumption. The second and third plot show the active power generated by genset 1

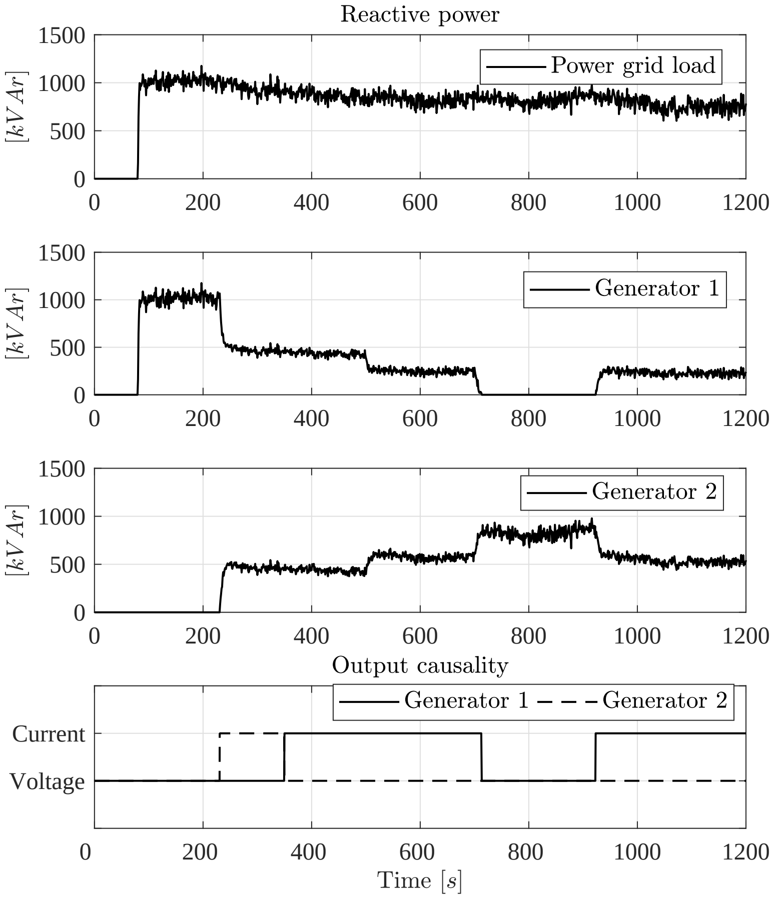


Figure 6.18: Reactive Power

and 2, respectively. From the two plots, it can be seen that the noise experienced by the gensets decreases when both gensets are active due to active load sharing. One can also see that the sum of generated active power equals the active power consumed by the power grid load. The last plot shows the output causality of the two gensets. Note that before genset 1 is stopped all the active power production is transferred to genset 2. This is one of the many functionalities implemented in the PMS.

Figure 6.18 shows the reactive power for the two gensets and the power grid load, as well as the genset output causalities. As can be seen in the figure, the results

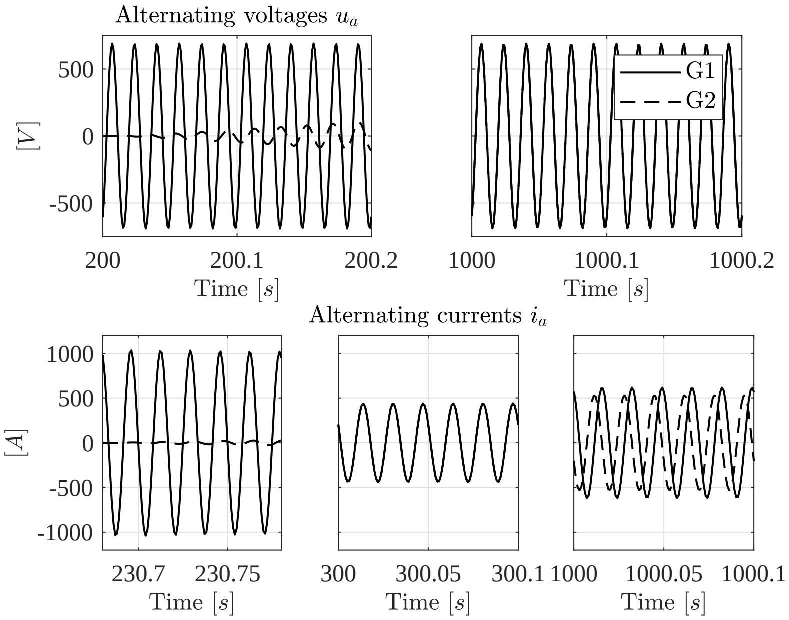


Figure 6.19: Voltages and currents, a, in the (a, b, c)-reference frame

are similar to the ones presented in Figure 6.17, the sum of the reactive power for the two gensets equals the reactive power in the power grid load, and that all the reactive power is transferred to genset 2 before genset 1 is stopped.

Figure 6.19 shows the alternating voltages u_a and currents i_a for the two generators in the chosen time ranges. The plot in the upper left-hand corner shows the u_a for the two generators when genset 2 is started, while the upper plot to the right shows the two voltages in a time range when both circuit breakers are closed. When both generators are connected to the grid, the alternating voltages should overlap, which the upper right-hand plot proves. The plot in the lower left-hand corner shows the currents i_a right after genset 2 is connected and load sharing is activated. This can be seen in the end of the plot, in which the current from genset 2 is starting to increase. The lower plot in the middle shows the two currents in an equal reactive power sharing settings, and the last plot shows a time range when the reactive power sharing is asymmetric.

6.4.5 System Evaluation

The simulation results show that the hybrid generator models seem to be stable, which is not that surprising according to the chosen solver time step. Also, the results show that switching between causalities do not affect the simulation results. This means that the integrator reset algorithm and the inheritance of initial conditions are done correctly in the model.

The control systems also seem to work properly, being able to both control the power grid frequency, the power grid rms voltage, phase synchronization and load sharing in a fast and stable manner despite being simple PID-based control laws. Such control laws are also used widely in the industry because they are simple to implement, simple to tune and quite robust as long as the controlled systems have strong linear characteristics in the controlled region or certain passivity properties.

The total simulation of $1200\,\mathrm{s}$ was solved in approximately $100\,\mathrm{s}$, giving a solving speed of approximately 12 times real-time. This indicates that the computational efficiency is being maintained for real-time application purposes without affecting the quality of the simulation results.

6.5 Conclusion

This chapter set out to present a generic framework for modeling and simulating marine power plants with weak power grids in transient operations. The generator model presented here is a hybrid model that has the ability to switch between outputs and inputs, which makes it suited for transient power plant operations such as genset start-up and shutdown, as well as phase synchronization control. This also enables the ability to test failure modes such as genset trippings due to the overload or overstepping of power grid frequency tolerances, but which has not been discussed in any detail here.

Two hybrid causality generator models were implemented in a small marine power plant as a case study. The differential causality in the voltage output generator model was solved by differentiation using a transfer function, and a generator parameter analysis was conducted based on the system eigenvalues. The current output model seemed to be difficult to solve using the Euler integration method, but a simulation setup that enabled the use of the Euler method in a larger scale simulation, such as the one performed, was proposed and tested through simulations.

An overall power plant control system was proposed through the control of the field voltages and the engine speeds by using simple but effective controllers for rms voltage, engine speed, phase synchronization and active- and reactive power sharing. The simulation results demonstrated that the total power plant model was stable, even though noise was added, and all transient effects of interest could be reflected through the simulation results. The results also demonstrated the equivalence between the $(d, q, 0)$-reference frame and the (a, b, c)-reference through comparison with well-known alternating generator characteristics. The simulation results are also comparable to the results obtained in [164], in which the simulation results were verified with experimental data. Also, the power plant itself showed good real-time conditions, being solved about 12 times faster than real-time.

In further work, the proposed power plant framework should be validated against experimental results. However, since the genset models are based on well-known equations, it is believed that the simulation results should converge to experimental results, as long as enough information about the control laws and the power management system is given. Also, a sophisticated outer control law for planning the total power plant operation, e.g. deciding when to start and stop a genset from power plant measurements, should be considered. This is out of scope here but it is believed that the power plant model presented in this case study gives a good foundations for such studies.

The marine power plant model derived in this chapter will be further used in Chapter 8 where it is implemented as one subsystem in a co-simulation of a marine vessel in DP-operations.

CHAPTER 7

Thrust Allocation Algorithm for Marine Vessels

This chapter is based on a restructured version of [40, see **P8** in section 1.6] where a thrust allocation algorithm based on *Model Predictive Control* (MPC) theory is derived. In contrast to the many contributions to thrust allocation algorithms in the literature, the derived algorithm is not based on azimuth angles, only on vector formulations, making the algorithm more linear and more simple to solve. The proposed algorithm is benchmarked against a commonly used one-step thrust allocation algorithm and a study of optimal cost function weights and optimization time horizon is performed. A thorough case-study including the vessel model derived in Chapter 5 is performed and different cost function weights are tested in simulations and discussed with focus on reduction in thrust force oscillations. As it turns out, the use of thruster biasing and actively compensating for fluctuations in thrust forces by the use of thruster azimuth angles reduce oscillations in the power consumption if the cost function weights are chosen properly. The proposed thrust allocation algorithm is to be further used in co-simulations in Chapter 8.

7.1 Introduction

In marine offshore operations, proper control of marine vessels and equipment are important both in order to complete the task at hand at the same time as keeping the costs at a minimum and maintaining a financial surplus. This must be done within the requirements set by the customers in order to maintain a good reputation such that new contracts can be made easier. On the other hand, marine operations tend to become more demanding, requiring higher precisions and special qualifications at the same time as environmental footprints have become more highlighted than before. Since every third party vendor put a lot of effort into research and development of their products, the largest potential of improvements of a vessel in specific operations is within system integration, namely how the vessel's equipment should be interfaced and controlled in order to obtain the best performance possible. One such important system integrator is the thrust allocation algorithm, which connects and transforms the global commanded thrust signals from the DP-controller to the propulsion system.

A lot of work has been done regarding thrust allocation for marine vessels, as well as in the aerospace industry. It has been growing interests in formulating the thrust allocation problems as Model Predictive Control (MPC) problems [175], thus, most often including thruster angles and absolute thrust in the problem formulation. In general, an MPC is an optimization based method for the feedback control of a system. Model predictive control is also known as a moving horizon control (MHC) and receding horizon control (RHC) because it optimizes over a given time horizon, as will be elaborated in section 7.2, and is often used for controlling slow dynamical systems [176]. A thorough survey of model predictive control theory and practice is given in [177], and in [178] where the recent developments and future promises of MPC is discussed. Hence, these topics will not be given much attention here. For a thorough introduction to both linear and non-linear MPC theory the reader is referred to [179] and [180], respectively.

A thrust allocation method with dynamic power consumption modulation for diesel-electric ships is presented in [181]. This algorithm leads to a more stable loading of the power plant for reduction in fuel consumption, in addition to reducing wear of the power plant, in comparison to standard thrust allocation algorithms. In [182] a robust control allocation for over-actuated ships is considered and verified by experiments with a model ship. Much attention is devoted to reducing the load variations on the power plant through proper thrust allocation designs using MPC in [183], where a thrust allocation algorithm including fuel consumption characteristics is presented, and [184], where the thrust allocation algorithm includes power management functionalities for reduction in frequency and load variations on the electric network. Also, in the field of aerospace, control allocation problems have been addressed in [185, 186], and are similar to the thrust allocation problem in the marine environment.

In general, the thrust allocation problem for a marine vessel can be solved explicitly [187] for non-rotatable actuators, as done in [188] and [189]. However, when considering rotatable thrusters, the thrust allocation problem becomes implicit, and is often not convex when only allowing thruster angles to have numerical values within certain regions [190]. However, in [189] and [191] an explicit method for solving the thrust allocation problem using rotatable thrusters and piecewise linear functions is proposed and can be used in order to include rotatable thrusters in one-step optimization problems as well. In industrial thrust allocation algorithms, the use of one-step optimization is prevalent. This, because of its simplicity as well as being generally fast to solve, which is an important requirement in real-time implementations.

In some applications the DP-controller is integrated into the thrust allocation optimization problem, as done in [192], in contrast to a stand-alone algorithm feeding the thrust allocation algorithm with global thrust commands. Also, the fidelity of DP-controllers span from model based control designs [193] and DP-controllers including advanced filters [194, 195] to simple PID-control based designs including rotational matrices [14]. A thorough survey of DP-control systems is given in [20].

In the closed control loop, consisting of the available measurements, the DP-controller, the thrust allocation algorithm and the propulsion system, there is often a filter as well. This filter takes the available position and orientation measurements and tries to filter out the fastest oscillatory environmental effects. Such filters could be based on system dynamics such as Kalman filters [196] and non-linear passive observers [148], that also include observer properties, or simple filters such as band-pass, band-stop or low-pass filters. Although suited filters are used, it is hard to filter out all the unwanted wave- and environmental effects without introducing a significant phase-lag, and these effects tend to be amplified through the DP-control system if real care is not taken. Hence, filtering properties in the thrust allocation algorithm are much appreciated.

In this chapter a non-angular MPC-based thrust allocation algorithm framework for maritime surface vessels in DP-operations, including an optimization horizon larger than the one-step method, is proposed, tested and benchmarked against a standard non-linear one-step thrust allocation algorithm. By including an optimization horizon of proper length it will be shown that the proposed thrust allocation algorithm can obtain filtering properties that not only reduce oscillatory environmental disturbances, but also maintain a low power consumption while keeping the vessel in position. Because of the new optimization problem formalism presented here, that excludes thruster angles in the problem formulation, the algorithm would also have good real-time properties even though having a significant optimization horizon length that increases the problem quadratically, as will be shown in various simulation results presented in this chapter. In contrast to the algorithms presented in the literature, the proposed algorithm is simple to formulate while not being subject to piecewise linearisation, multiple shooting strategies or additional functionalities for ensuring faster convergence, which saves development time.

The proposed thrust allocation algorithm does not contain any information about fuel consumption in the power plant, in contrast to the already mentioned literatures. However, it takes aim at reducing the thrust commands and the thrust command oscillations in order to both smooth the power consumption as well as keeping the loading of the power plant as low as possible. This, because the consumed power is approximately proportional to the generated thrust to the power of $\frac{3}{2}$, see [192].

The main case study presented in this chapter, which includes a high fidelity vessel model with all relevant auxiliary systems as the one presented in Chapter 5, provides reasonable and realistic results which is crucial when optimizing system integration such as the interaction between the measurements, the filter, the DP-controller, the thrust allocation algorithm and the local thruster controllers, as done here, where the thrust allocation algorithm is the main integrator between the different systems. Different key performance indicators (KPI's) such as total power consumption, position and orientation errors, and power spectral density (PSD) analysis of the thrust command outputs, are used to compare different sets of cost function weights in the thrust allocation algorithm. This type of thorough study is not very prevalent in the literature. Hence, it has been devoted much attention here.

When working with non-linear MPC problems, often additional work must be put into assuring global convergence of the optimization problem, such as adding soft constraints [197]. However, such soft constraints are not needed in the proposed algorithm. A non-angular vector formulation contains the same amount of information as an angle-amplitude formulation, but the drawbacks are that the thrust angles and the thrust amplitudes must be calculated from the vectors before being fed to the propulsion system, and that the thrust rates and the thrust constraints tend to become a bit conservative. However, as will be shown in section 7.2.4, calculating the thrust angles and amplitudes are simple and explicit procedures.

The reason for using an MPC-algorithm instead of a one-step optimization algorithm is primarily to be able to work with thrust rates since such algorithms often run around $1\,Hz$. Then, it is possible to integrate the optimal rates outside the thrust allocation algorithm in order to obtain smooth thrust commands. When using a one-step algorithm, this is much more difficult since the algorithm has only one sample in the horizon before reaching the reference values, which compromises reducing the rates. A longer optimization horizon also enables planning of the thrust commands on a future time horizon, in contrast to a one-step algorithm. This effect is studied in a benchmarking test that compares the proposed algorithm and a one-step algorithm in a simple DP-operation case.

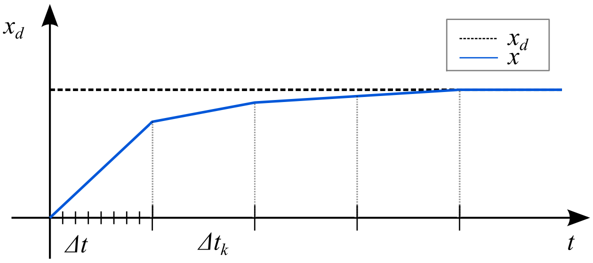


Figure 7.1: Sketch of solution from the thrust allocation algorithm where x_d is the desired thrust and x is the corresponding output from the thrust allocation algorithm

7.1.1 Structure of Chapter

This Chapter is structured as follows: First, the proposed thrust allocation algorithm is presented in section 7.2 before being benchmarked against a one-step thrust allocation algorithm in a DP-operation simulation case in section 7.3. Secondly, simplified preliminary case studies including only the thrust allocation algorithm are initiated in section 7.4 in order to highlight how the characteristics of the proposed thrust allocation algorithm are affected by different cost function weights and optimization horizon lengths. In the end, a main case study is presented in section 7.5 and shows how different choices of cost function weights affect the overall performance of a vessel in DP-operation excited by current and irregular sea states.

7.2 MPC-based Thrust Allocation Algorithm

In general, solving optimization problems, such as MPC problems, often tend to become computationally demanding and the optimization problem is often implemented with discrete dynamics and with a fixed number of samplings in the horizon. Here, the number of steps in the horizon, K, is given as

$$K = \text{floor}\left(\frac{T}{\Delta t_k}\right) \tag{7.1}$$

where T is the length of the time horizon treated in the optimization and Δt_k is the length of each sample in the horizon, as well as the time step between each optimization solver call. Hence, the thrust optimization is performed with a frequency $f_k = \Delta t_k^{-1}$, outputting thrust rates that can be integrated between each optimization with a time step Δt. Figure 7.1 gives an overview of how these time steps relate to each other.

Before presenting the proposed thrust allocation algorithm, a few definitions are needed. A thrust-vector representation is to be used instead of the traditional

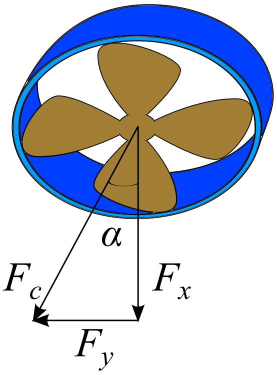


Figure 7.2: Thrust from propulsor given as both vector representation and amplitude-angle representation

amplitude-angle representation for rotatable thrusters in order to reduce non-linearities in the optimization problem related to the thruster angles. This means that each rotatable thruster can be represented by two thrust vectors, F_x and F_y, as shown in Figure 7.2. These two thrust vectors include the same amount of information as the traditional amplitude-angle representation, given as F_c and α, respectively, in the figure. Relations between the amplitude-angle based thrust representation and the vector representation are given as

$$F_c = \sqrt{F_x^2 + F_y^2} \tag{7.2a}$$

$$\alpha = \arctan\left(\frac{F_y}{F_x}\right) \tag{7.2b}$$

Also, the sign convention for the thrust vectors are set according to the body-fixed reference frame of the vessel, as shown in Figure 7.3. This means that a positive thrust in F_x moves the ship forward and a positive thrust in F_y moves the ship to starboard. For a thruster with fixed azimuth angle, such as a bow thruster, the thrust amplitude is used along with the static thrust angle, α_s.

When having N number of thrusters, the global thrust vectors in surge, sway and yaw, denoted as $X_N(t_k)$, $Y_N(t_k)$ and $Mz_N(t_k)$, respectively, for time step t_k, can

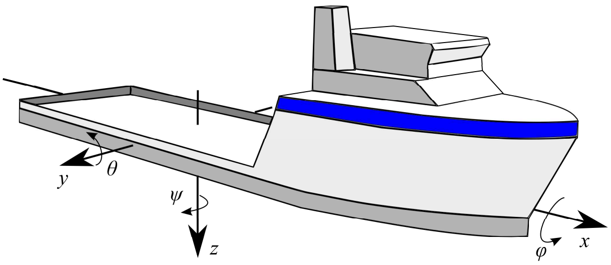


Figure 7.3: Body-fixed reference frame of the vessel

be calculated as

$$X_N(t_k) = \sum_{i=1}^{N} F_{ix}(t_k) \tag{7.3a}$$

$$Y_N(t_k) = \sum_{i=1}^{N} F_{iy}(t_k) \tag{7.3b}$$

$$Mz_N(t_k) = -\sum_{i=1}^{N} [F_{ix}(t_k)y_i - F_{iy}(t_k)x_i] \tag{7.3c}$$

where $(x_i,\, y_i,\, z_i)$ is the position of thruster i. For convenience, these total thrust contributions are given in vector form for time step t_k as

$$\boldsymbol{x}(t_k) = \begin{bmatrix} X_N(t_k), & Y_N(t_k), & Mz_N(t_k) \end{bmatrix}^\top \tag{7.4}$$

and the corresponding desired thrust given by a potential DP-controller is given as $\boldsymbol{x}_d(t_k)$. Note that the thrust command is assumed constant during the whole horizon. This can be argued for when having a DP-controller that provides the desired thrust command, containing integration effects, in combination with a small Δt_k. However, this assumption should be verified through simulations.

A bounded variable f is said to be defined in the range $f \in [\underline{f}, \overline{f}]$ such that

$$\begin{aligned} \min(f) &= \underline{f} \\ \max(f) &= \overline{f} \end{aligned} \tag{7.5}$$

Also, the bound vector is defined as

$$\underline{\overline{f}} = [\underline{f}, \overline{f}] \tag{7.6}$$

An absolute value, F_c, of two thrust vectors F_x and F_y, as in (7.2a), is said to be signed if it is negative when the thruster is reversed and positive if not, and is denoted as $F_c^{\pm}$. Also note that $k \in [1, .., K]$ is the sampling number of the MPC horizon and is used as a place-holder for the time step k, t_k, such that $[t_0, t_1, .., t_k, .., t_K]$ and $t_K = T$.

In the thrust allocation algorithm presented in this chapter, physical limitations need to be addressed in the algorithm. This is done through constraints in the optimization problem, and is elaborated in the following.

7.2.1 Constraints

In an optimization problem, constraints are used in order to assure that the optimal solution is realistic and possible to implement in a realistic system. Often these physical constraints are related to maximal values and rates, e.g. taking into consideration the maximal capacity of the system and how fast the system states are able to change.

Here, all constraints will be presented in continuous time even though being implemented in discrete time in section 7.2.3. The thrust vectors are assumed to be represented as states in the thrust allocation algorithm in order to include rate-limitations and to be able to calculate new thrust commands between each optimization call, Δt_k. Hence, the thrust vectors are expressed as

$$\frac{d}{dt} F_x(t) = u_x(t) \tag{7.7a}$$

$$\frac{d}{dt} F_y(t) = u_y(t) \tag{7.7b}$$

for a rotatable thruster, where u_x and u_y are control variables. Note that these differential equations are not comparable to the differential equations describing a thruster, it is only an implementation for enabling rate limitations for the thrust commands. If the thruster angle is fixed, only one control variable is needed, u, and u_x and u_y can be calculated from u and the static thruster angle α_s. Then, the thrust rate constraints for thruster i can be set as

$$\underline{u_{ix}} \leqslant u_{ix}(t) \leqslant \overline{u_{ix}} \tag{7.8a}$$

$$\underline{u_{iy}} \leqslant u_{iy}(t) \leqslant \overline{u_{iy}} \tag{7.8b}$$

where $[\underline{u_{ix}}, \overline{u_{ix}}]$ and $[\underline{u_{iy}}, \overline{u_{iy}}]$ are the allowed regions for the thrust vector rates. For rotatable thrusters the allowed regions can be calculated between each optimization, or simply assumed given as

$$\overline{\underline{u_{ix}}} = \overline{\underline{u_{iy}}} = \frac{1}{\sqrt{2}} \overline{\underline{u_{ic}}} \tag{7.9}$$

and for fixed thrusters

$$\overline{\underline{u_{ix}}} = \overline{\underline{u_{ic}}} \cos(\alpha_{is}) \tag{7.10a}$$

$$\overline{\underline{u_{iy}}} = \overline{\underline{u_{ic}}} \sin(\alpha_{is}) \tag{7.10b}$$

where $[\underline{u_{ic}}, \overline{u_{ic}}]$ is the allowed thrust rate region for F_{ic}. Similarly, the maximal capacity constraint for thruster i can be formulated as

$$\underline{F_{ix}} \leqslant F_{ix}(t) \leqslant \overline{F_{ix}} \tag{7.11a}$$

$$\underline{F_{iy}} \leqslant F_{iy}(t) \leqslant \overline{F_{iy}} \tag{7.11b}$$

where $[\underline{F_{ix}}, \overline{F_{ix}}]$ and $[\underline{F_{iy}}, \overline{F_{iy}}]$ are the allowed thrust vector regions in each direction. As for the thrust rates, the allowed thrust regions for rotatable thrusters can be calculated between each optimization, or simply assumed given as

$$\overline{\underline{F_{ix}}} = \overline{\underline{F_{iy}}} = \frac{1}{\sqrt{2}} \overline{\underline{F_{ic}}} \tag{7.12}$$

and for fixed thrusters

$$\overline{\underline{F_{ix}}} = \overline{\underline{F_{ic}}} \cos(\alpha_{is}) \tag{7.13a}$$

$$\overline{\underline{F_{iy}}} = \overline{\underline{F_{ic}}} \sin(\alpha_{is}) \tag{7.13b}$$

where $\overline{\underline{F_{ic}}} = [\underline{F_{ic}}, \overline{F_{ic}}]$ is the allowed thrust region for F_{ic}. Note that the rate constraints and the maximal capacity constraints in (7.9) and (7.12), respectively, are a bit conservative for the rotatable thrusters. However, as will be seen in section 7.5, much focus will be given to reducing power consumption and keeping the thruster rates as low as possible to avoid large accelerations. Then, conservative rate constraints will have a minor impact on the optimal solution.

When adding the turning rate constraint on each rotatable thruster, e.g. $\frac{d}{dt}\alpha_i$, the orientation and the rate of the thrusters are not directly part of the general vector representation. However, the turning rate for thruster i can be calculated by differentiating (7.2b). Thus,

$$\begin{aligned} \dot{\alpha}_i(t) &= \frac{d}{dt} \arctan\left(\frac{F_{iy}(t)}{F_{ix}(t)}\right) \\ &= \frac{u_{iy}(t)F_{ix}(t) - u_{ix}(t)F_{iy}(t)}{F_{ix}(t)^2 + F_{iy}(t)^2} \end{aligned} \tag{7.14}$$

By defining the allowed angle rate region as $[\underline{\dot{\alpha}_i}, \overline{\dot{\alpha}_i}]$ for a thruster i, the rate constraints can be expressed as

$$u_{iy}(t)F_{ix}(t) - u_{ix}(t)F_{iy}(t) \leqslant \overline{\dot{\alpha}_i}(t)(F_{ix}(t)^2 + F_{iy}(t)^2), \tag{7.15a}$$

$$\underline{\dot{\alpha}_i}(t)(F_{ix}(t)^2 + F_{iy}(t)^2) \leqslant u_{iy}(t)F_{ix}(t) - u_{ix}(t)F_{iy}(t) \tag{7.15b}$$

The last constraints to be added are the initial conditions to the differential states, namely the thrust vectors. Hence, for a thruster i the initial conditions are given as constraints as

$$F_{ix}(0) := F_{ix,0} \tag{7.16a}$$

$$F_{iy}(0) := F_{iy,0} \tag{7.16b}$$

where $F_{ix,0}$ and $F_{iy,0}$ are either the previously obtained states or measurements. Along with constraints, cost functions are added to reflect the chosen optimal thrust. These cost functions are elaborated in the following.

7.2.2 Cost Functions

For such optimization problems as MPC's, it would be beneficial for the solving procedure if the cost functions are convex. This can be obtained by using quadratic cost functions, e.g. $\boldsymbol{z}^\top \boldsymbol{Q} \boldsymbol{z}$, where $\boldsymbol{z} \in \mathcal{R}^N$ is a vector and $\boldsymbol{Q} \in \mathcal{R}^{N \times N}$ is a weighting matrix, typically diagonal. Also, in some cases linear costs are added in order to improve convergence when the variables in the cost functions become small. However, this has not been deemed necessary here.

Since a time horizon with a fixed number of samplings is used in the thrust allocation algorithm, two sets of cost functions should be used; one intermediate cost function and one end cost function. Typically, the end cost function would represent the costs from the infinite time horizon that are neglected when assuming a finite time horizon, while the intermediate cost function, in addition to representing the main objective, would include costs that affect how the main objective is obtained.

Starting with the main objective, the quadratic intermediate cost function for time step k can be expressed as

$$[\boldsymbol{x}(k) - \boldsymbol{x}_d(k)]^\top \boldsymbol{Q}_k [\boldsymbol{x}(k) - \boldsymbol{x}_d(k)] \tag{7.17}$$

where k is the sampling number, $\boldsymbol{Q}_k \in \mathcal{R}^{3\times3}$ is a diagonal weight matrix and $\boldsymbol{x}(k), \boldsymbol{x}_d(k) \in \mathcal{R}^3$ are the global thrust vector and global demanded thrust vector,

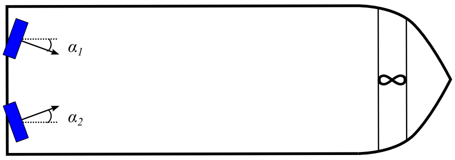


Figure 7.4: Biasing of main thrusters placed at the stern

respectively. Similarly, the end cost function can be expressed as

$$[\boldsymbol{x}(T) - \boldsymbol{x}_d(T)]^\top \boldsymbol{Q}_T \, [\boldsymbol{x}(T) - \boldsymbol{x}_d(T)] \tag{7.18}$$

where $\boldsymbol{Q}_T \in \mathcal{R}^{3\times3}$. Rate costs are assumed negligible in the end cost function since all large accelerations and rates are largest in the first samples in the time horizon and the thrust commands from the DP-controller are assumed constant during the entire time horizon.

In addition to the main objective cost function, rate costs and magnitude costs are added to the thrust control variables and the thrust vectors, respectively. Hence, the rate costs for N number of thrusters for time step k are given as

$$\boldsymbol{u}(k)^\top \boldsymbol{Q}_u \boldsymbol{u}(k) \tag{7.19}$$

where $\boldsymbol{u}(k) \in \mathcal{R}^{2N}$,

$$\boldsymbol{u}(k) = \left[\begin{array}{ccccc} u_{1x}(k), & u_{1y}(k), & ..., & u_{Nx}(k), & u_{Ny}(k) \end{array} \right]^\top \tag{7.20}$$

and $\boldsymbol{Q}_u \in \mathcal{R}^{2N\times2N}$ is a diagonal weight matrix. The magnitude costs are given similarly,

$$\boldsymbol{F}(k)^\top \boldsymbol{Q}_F \boldsymbol{F}(k) \tag{7.21}$$

where $\boldsymbol{F}(k) \in \mathcal{R}^{2N}$,

$$\boldsymbol{F}(k) = \left[\begin{array}{ccccc} F_{1x}(k), & F_{1y}(k), & ..., & F_{Nx}(k), & F_{Ny}(k) \end{array} \right]^\top \tag{7.22}$$

and $\boldsymbol{Q}_F \in \mathcal{R}^{2N\times2N}$. In addition, if the ship has two main thrusters placed at the stern, it would in some cases be of interest to cancel them against each other or to store available thrust in order to obtain a faster response, as shown in Figure 7.4, and is often referred to as thruster biasing. Note that thruster biasing is also used in some applications for singularity avoidance, but this is not the case here. Thruster biasing can be included in the optimization problem by changing the cost function given in (7.21) to

$$[\boldsymbol{F}(k) - \boldsymbol{\delta}(X_d(k))]^\top \boldsymbol{Q}_F [\boldsymbol{F}(k) - \boldsymbol{\delta}(X_d(k))] \tag{7.23}$$

where $\boldsymbol{\delta}(X_d(k)) \in \mathcal{R}^{2N}$,

$$\boldsymbol{\delta}(X_d(k)) = \begin{bmatrix} 0, & \delta_{1y}(X_d(k)), & ..., & 0, & \delta_{Ny}(X_d(k)) \end{bmatrix}^\top \tag{7.24}$$

and

$$\delta_{iy}(X_d(k)) = |X_d(k)| r_i \arctan(\alpha_{ib}) \tag{7.25}$$

for the two main thrusters placed at the stern, where α_{ib} is the biasing angle for thruster i and r_i is the fraction of the total maximal thrust in surge thruster i can provide.

The last intermediate cost function to be added is perhaps the most important one after the main objective, namely a cost function that makes thruster biasing optimal. By adding such a cost function to the optimization problem, the total thrust allocation algorithm has the ability to obtain the main objective without accelerating or de-accelerating the thrusters too fast. This would also benefit the goal of obtaining a smooth power demand by the thrusters. This cost function is given as

$$\boldsymbol{u}(k)^\top \boldsymbol{b}(\boldsymbol{F}(k))^\top \boldsymbol{Q}_{uF} \boldsymbol{b}(\boldsymbol{F}(k)) \boldsymbol{u}(k) \tag{7.26}$$

where $\boldsymbol{b}(\boldsymbol{F}(k)) \in \mathcal{R}^{2N \times 2N}$,

$$\boldsymbol{b}(\boldsymbol{F}(k)) = \begin{bmatrix} \frac{F_{1x}(k)}{F_{1c}(k)+\epsilon} & 0 & \cdots & 0 \\ 0 & \frac{F_{1y}(k)}{F_{1c}(k)+\epsilon} & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \frac{F_{Ny}(k)}{F_{Nc}(k)+\epsilon} \end{bmatrix} \tag{7.27}$$

and ϵ is a small number added in order to avoid dividing by zero and $F_{ic}(k)$ is given similarly as in (7.2a),

$$F_{ic}(k) = \sqrt{F_{ix}(k)^2 + F_{iy}(k)^2} \tag{7.28}$$

One could perhaps argue for the fact that the cost function in (7.19) and and the cost function in (7.26) are similar and can be combined into one cost function. However, since the diagonal terms in (7.27) scale the cost function weights, and since the cost for the thrust rate u_{ip} is zero when $F_{ip} = 0$, where $p \in (x, y)$, the diagonal terms in (7.27) must then be updated to $c_i + \frac{F_{ip}}{F_{ic}+\epsilon}$ where $c_i > 0$ is a scaling parameter reflecting the difference between $\boldsymbol{Q}_u(i,i)$ and $\boldsymbol{Q}_{uF}(i,i)$. Hence, since there does not exist any computational advantages of combining (7.19) and (7.26), they are kept separated.

It is also possible to add cost functions for minimizing thruster-thruster interactions for thrusters placed side by side, meaning that if one thruster is facing the other it will reduce the efficiency of that thruster since its wake will affect the other thruster. This could for example be included as a function of $\left|\frac{F_{iy}}{F_{ix}}\right|$, since when F_{iy} becomes large in comparison to F_{ix}, the thruster angle is approaching a sway oriented direction. However, such cost functions are not included here.

The total MPC formulation of the thrust allocation algorithm is summarized and put together in the following.

7.2.3 Total MPC Formulation

The constraints and cost functions have been presented in section 7.2.1 and 7.2.2, respectively. The constraints have been derived in continuous time and the cost functions have been given for given time steps k, except for the end cost function in (7.18). By combining all the constraints and the cost functions, the total MPC formulation of the thrust allocation problem can be formulated as

$$\min_{\boldsymbol{u}\in\mathcal{R}} \; J(\boldsymbol{x}, \boldsymbol{x}_d, \boldsymbol{u}, \boldsymbol{F}, \boldsymbol{\delta}(X_d), T) \tag{7.29a}$$

$$\text{subject to } \forall i \in [1, ..., N] \text{ and } \forall k \in [1, ..., K]$$

$$F_{ix}(0) := F_{ix,0} \tag{7.29b}$$

$$F_{iy}(0) := F_{iy,0} \tag{7.29c}$$

$$F_{ix}(k) := F_{ix}(k-1) + u_{ix}(k)\Delta t_k \tag{7.29d}$$

$$F_{iy}(k) := F_{iy}(k-1) + u_{iy}(k)\Delta t_k \tag{7.29e}$$

$$\overline{\boldsymbol{g}_i(k)} \leqslant \boldsymbol{0} \tag{7.29f}$$

$$-\underline{\boldsymbol{g}_i(k)} \leqslant \boldsymbol{0} \tag{7.29g}$$

where the total cost function $J(\cdot)$ is given as

$$\begin{aligned} J(\cdot) = &\sum_{k=1}^{T-1} \left[\boldsymbol{x}(k) - \boldsymbol{x}_d(k)\right]^\top \boldsymbol{Q}_k \left[\boldsymbol{x}(k) - \boldsymbol{x}_d(k)\right] \\ &+ \sum_{k=1}^{T-1} \boldsymbol{u}(k)^\top \boldsymbol{Q}_u \boldsymbol{u}(k) \\ &+ \sum_{k=1}^{T-1} [\boldsymbol{F}(k) - \boldsymbol{\delta}(X_d(k))]^\top \boldsymbol{Q}_F [\boldsymbol{F}(k) - \boldsymbol{\delta}(X_d(k))] \\ &+ \sum_{k=1}^{T-1} \boldsymbol{u}(k)^\top \boldsymbol{b}(\boldsymbol{F}(k))^\top \boldsymbol{Q}_{uF} \boldsymbol{b}(\boldsymbol{F}(k)) \boldsymbol{u}(k) \\ &+ \left[\boldsymbol{x}(T) - \boldsymbol{x}_d(T)\right]^\top \boldsymbol{Q}_T \left[\boldsymbol{x}(T) - \boldsymbol{x}_d(T)\right] \end{aligned} \tag{7.30}$$

and the inequality constraint vector function $\overline{\underline{\boldsymbol{g}_i(k)}}$ is given as

$$\overline{\underline{\boldsymbol{g}_i(k)}} = \begin{bmatrix} u_{ix}(k) - \overline{\underline{u_{ix}}} \\ u_{iy}(k) - \overline{\underline{u_{iy}}} \\ F_{ix}(k) - \overline{\overline{\underline{F_{ix}}}} \\ F_{iy}(k) - \overline{\overline{\underline{F_{iy}}}} \\ u_{iy}(k)F_{ix}(k) - u_{ix}(k)F_{iy}(k) - \overline{\underline{\dot{\alpha}_i}}F_{ic}(k) \end{bmatrix} \tag{7.31}$$

and $\Delta t_k = t(k) - t(k-1)$ is the time between two sampling intervals. Note that (7.29d) and (7.29e) are the discrete implementations of (7.7a) and (7.7b), respectively. Also note that (7.29g) is a restriction making sure that the lower bound of $\boldsymbol{g}_i(k)$ is larger than zero.

The outputs from the thrust allocation algorithm are $F_{ic}^{\pm}$ and α_i for rotatable thrusters and F_{ic} for fixed ones such as tunnel thrusters. Since the thruster angles for the rotatable thrusters are not directly included in the optimization problem formulation, the angles, α_i, need to be calculated afterwards from the optimal MPC output. The same goes for the signed thrust amplitudes $F_{ic}^{\pm}$. These calculations are elaborated in the following.

7.2.4 Implementation

Usually, thrusters are controlled based on thruster angles and signed thrust amplitudes. When using a thrust vector representation to describe the thrust forces generated by each thruster instead of thruster angles and signed thrust amplitudes, some logics must be implemented in order to obtain the desired control signals for the thrusters after the optimization algorithm. In general, logics must be implemented in order to count the number of rotations a thruster goes through, in order to produce a continuous thruster angle signal, and logics that determine whether the MPC rotates or reverses a given thruster.

By using $\arctan 2(\cdot)$ instead of $\arctan(\cdot)$ in (7.2b), one can count the number of rotations by comparing the previously calculated thruster angle with the current one. This, in order to assure that the angle commands do not contain discontinuities. This procedure is summarized in Algorithm 7.2.

In the algorithm, n_i is the rotation counter and m_i is another counter used for calculating whether the thrust allocation algorithm rotates a thruster or reverses the corresponding thrust. This can be evaluated after running Algorithm 7.2 since the thruster angle signal doesn't contain any discontinuities related to the trigonometric function. Hence, the logics needed to determine whether a thruster is rotated or reversed by the MPC may be implemented as in Algorithm 7.3.

In the algorithm, s_i is a sign variable, $s_i \in [-1, 1]$, and ϵ is a small number, $\epsilon > 0$.

Algorithm 7.2 Counting thruster rotations

```
procedure RotationCount(·)
    $\alpha_i(t) = 2\pi n_i + \pi m_i + \arctan 2(F_{iy}(t), F_{ix}(t))$
    if $|\alpha_i(t) - \alpha_i(t-1)| \geqslant 2\pi$ then
        if $\alpha_i(t) - \alpha_i(t-1) < 0$ then
            $n_i = n_i + 1$
        else
            $n_i = n_i - 1$
        end if
        $\alpha_i(t) = 2\pi n_i + \pi m_i + \arctan 2(F_{iy}(t), F_{ix}(t))$
    end if
end procedure
return $\alpha_i(t)$
```

Algorithm 7.3 Rotating v.s. reversing thruster

```
procedure RotateOrReverse(·)
    $\alpha_i(t) = 2\pi n_i + \pi m_i + \arctan 2(F_{iy}(t), F_{ix}(t))$
    $F_{ic}^{\pm}(t) = s_i\sqrt{F_{ix}(t)^2 + F_{iy}(t)^2}$
    if $|\alpha_i(t) - \alpha_i(t-1)| > \dot{\alpha}_{i,max}\Delta t$ and $|F_{ic}^{\pm}(t)| \leqslant \epsilon$ then
        if $|\alpha_i(t)| - |\alpha_i(t-1)| < 0$ then
            $m_i = m_i + 1$
        else
            $m_i = m_i - 1$
        end if
        $s_i = -s_i$
        $\alpha_i(t) = 2\pi n_i + \pi m_i + \arctan 2(F_{iy}(t), F_{ix}(t))$
        $F_{ic}^{\pm}(t) = s_i\sqrt{F_{ix}(t)^2 + F_{iy}(t)^2}$
    end if
end procedure
return $\alpha_i(t), F_{ic}^{\pm}(t)$
```

In general, this algorithm checks if the rate constraint for the thruster angle is violated, and if the thrust magnitude is small, then the thruster has been reversed according to the optimization algorithm. The total thrust allocation algorithm including Algorithm 7.2 and 7.3 has been implemented in the C++ library ACADO [198] and solved by the qpoases library [199].

Even if the constraints and the cost functions are implemented as in (7.29), the characteristics of the optimization are not necessarily fixed. By tuning the weighting matrices different characteristics of the optimized thruster commands may be obtained, which will be shown in section 7.4 and 7.5. Before analysing how the tuning affects the performance of the proposed thrust allocation algorithm, the thrust allocation algorithm is to be compared to a more standard thrust allocation algorithm in a benchmarking test.

7.3 Benchmarking

In order to test the proposed thrust allocation algorithm a one-step algorithm, similar to the one presented in [190], is to be used for comparison. The reason why this algorithm is called a one-step algorithm is because the optimization horizon consists of only one point. Here, the horizon time is T=1 s, having only one sample. It might seem a bit unfair to compare a one-step optimization algorithm to an MPC algorithm, however, such one-step optimization algorithms are often used in industrial applications and, hence, suited to be used for comparison. It is expected that the proposed thrust allocation algorithm will outperform the one-step algorithm regarding reducing thrust rates and thruster angle rates, while reducing the total power consumption. However, it is also expected that the one-step algorithm will be faster than the proposed algorithm. Hence, the total energy consumption for the two algorithms, as well as the mean computational time, are to be compared as two of the key-parameters in this study.

The one-step optimization thrust allocation algorithm used for comparison is given as

$$\underset{\boldsymbol{F}_c,\boldsymbol{\alpha}_c\in\mathcal{R}}{min} J(\boldsymbol{x}, \boldsymbol{x}_d, \boldsymbol{\Delta\alpha}, \boldsymbol{F}_c, \boldsymbol{\Delta F}_c) \tag{7.32a}$$

subject to

$$\underline{\boldsymbol{F}} \leqslant \boldsymbol{F}_c \leqslant \overline{\boldsymbol{F}} \tag{7.32b}$$

$$\underline{\Delta\boldsymbol{F}} \leqslant \Delta\boldsymbol{F}_c \leqslant \underline{\Delta\boldsymbol{F}} \tag{7.32c}$$

$$\underline{\Delta\boldsymbol{\alpha}} \leqslant \Delta\boldsymbol{\alpha}_c \leqslant \overline{\Delta\boldsymbol{\alpha}} \tag{7.32d}$$

where

$$\begin{aligned} J(\cdot) = {} & (\boldsymbol{x}-\boldsymbol{x}_d)^\top \boldsymbol{Q}_e(\boldsymbol{x}-\boldsymbol{x}_d) + \Delta\boldsymbol{\alpha}^\top \boldsymbol{Q}_{\Delta\alpha}\Delta\boldsymbol{\alpha} \\ & + \boldsymbol{F}_c^\top \boldsymbol{Q}_F \boldsymbol{F}_c + \Delta\boldsymbol{F}_c^\top \boldsymbol{Q}_{\Delta F}\Delta\boldsymbol{F}_c \end{aligned} \tag{7.33}$$

and $\boldsymbol{x}$ is as defined in (7.4), $\boldsymbol{x_d}$ is the corresponding reference given by the DP-controller, $\Delta\boldsymbol{\alpha}$ is a vector containing the thruster angle rates for the two thrusters placed at the stern of the vessel, and is calculated as the difference between the previous output of the algorithm and the current output of the algorithm. $\boldsymbol{F}_c$ is a vector containing the three thrust amplitudes for the thrusters, $\Delta\boldsymbol{F}_c$ is the thrust

Table 7.1: Main parameters in vessel model

Parameter	Description	Value
L	Length of ship	107 m
B	Width of ship	22 m
D	Draught of ship	5 m
$P_{m,max}$	Power saturation main thrusters	3.5 MW
$P_{b,max}$	Power saturation bow thruster	3.5 MW
v_{cN}	Northward current	-0.1 m/s

amplitude rates for the three thrusters, calculated the same way as the thruster angle rates, $\overline{\boldsymbol{F}_c}$, $\overline{\Delta\boldsymbol{F}_c}$ and $\overline{\Delta\boldsymbol{\alpha}_c}$ are the limit vectors for the vectors $\boldsymbol{F}_c$, $\Delta\boldsymbol{F}_c$ and $\Delta\boldsymbol{\alpha}_c$, respectively. $\boldsymbol{Q}_e$ is the cost matrix for the error in global thrust, $\boldsymbol{Q}_{\Delta\alpha}$ is the cost matrix for the thruster angle rates, $\boldsymbol{Q}_F$ is the cost matrix for the thrust amplitudes and $\boldsymbol{Q}_{\Delta F}$ is the cost matrix for the thrust amplitude rates.

To simulate the performance of the two thrust allocation algorithms, the vessel model derived in [38] (see Chapter 5) is to be utilized. The thruster configuration for the vessel is the same as shown in Figure 7.4, e.g. two main thrusters symmetrically placed at the stern and one tunnel thruster in the bow, and the produced thrust is assumed measurable. Note that the main thrusters are rotatable and the bow thruster is fixed and produces thrust only in the sway direction. The main vessel parameters and thruster parameters are given in Table 7.1 and 7.2, respectively. Note that Table 7.2 sets the restrictions in (7.32b)-(7.32d).

A total overview of the simulation model is given in Figure 7.5. In the figure, $\boldsymbol{y}_m$ refers to the measurement vector including measurements of the vessel's position and heading, $\boldsymbol{y}_m^f$ is the filtered measurement vector, $\dot{\boldsymbol{y}}_m^f$ is the rate of the filtered measurement vector, $\boldsymbol{y}_d$ is the vector of the desired position and heading, $\dot{\boldsymbol{y}}_d$ is the corresponding rate vector, and τ_i is the thrust output from thruster i. Note that the thrust contributions from each thruster is transformed into global thrust contributions,

$$\boldsymbol{\tau}_g = \boldsymbol{H}(\alpha_1, \alpha_2)\boldsymbol{\tau} \tag{7.34}$$

where $\boldsymbol{\tau}_g \in \mathcal{R}^6$ is the global thrust vector, $\boldsymbol{H}(\cdot) \in \mathcal{R}^{6\times 3}$ is the thrust allocation matrix and $\boldsymbol{\tau} = [\tau_1,\ \tau_2,\ \tau_3]^\top$.

In order to compare the two different thrust allocation algorithms a manoeuvring test is designed and is elaborated in the following, along with the choice of cost function weights in the two thrust optimization formulations.

Table 7.2: Main parameters describing the thruster configuration as given in Figure 7.4

Thruster	Parameter	Description	Value
Main thruster port side (PS) and starboard (SB)	i	Thruster ID	PS :1, SB: 2
	(x,y)	Thruster position $[m]$	$(-45,\mp 7)$
	$\underline{F_{ic}}$	Min. thrust force $[N]$	-1000000.0
	$\overline{F_{ic}}$	Max. thrust force $[N]$	1000000.0
	$\underline{u_{ic}}$	Min. thrust rate $[\frac{N}{s}]$	-1000.0
	$\overline{u_{ic}}$	Max. thrust rate $[\frac{N}{s}]$	1000.0
	$\underline{\dot{\alpha}_i}$	Min. angular rate $[\frac{\circ}{s}]$	-10.0
	$\overline{\dot{\alpha}_i}$	Max. angular rate $[\frac{\circ}{s}]$	10.0
	α_{ib}	Bias angle $[^\circ]$	0.0
Bow thruster	i	Thruster ID	3
	(x,y)	Thruster position $[m]$	$(53,0)$
	$\underline{F_{ic}}$	Min. thrust force $[N]$	-1000000.0
	$\overline{F_{ic}}$	Max. thrust force $[N]$	1000000.0
	$\underline{u_{ic}}$	Min. thrust rate $[\frac{N}{s}]$	-1000.0
	$\overline{u_{ic}}$	Max. thrust rate $[\frac{N}{s}]$	1000.0

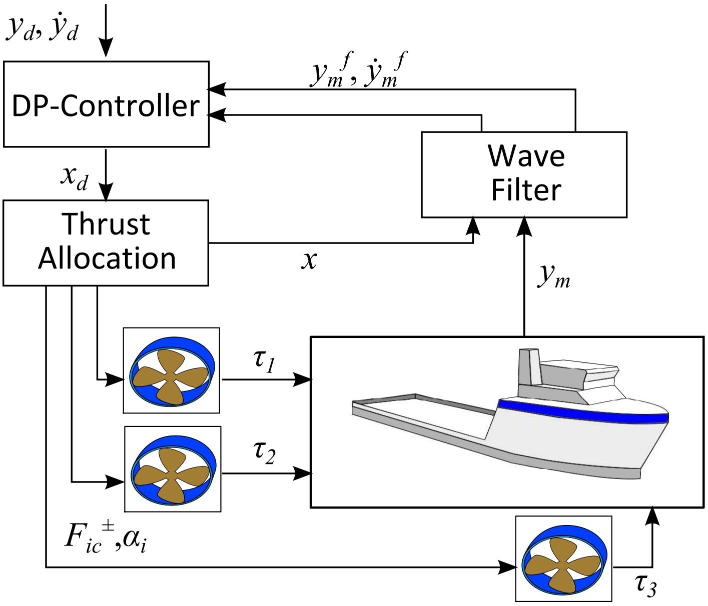


Figure 7.5: Simulation setup

Table 7.3: Cost function weights in the proposed thrust allocation algorithm and the one-step trust allocation algorithm. (i,i) denotes the entire diagonal of the matrix

MPC		**One-Step**	
$\boldsymbol{Q}_k(i,i)$	10.0	$\boldsymbol{Q}_c(i,i)$	100.0
$\boldsymbol{Q}_T(i,i)$	100.0	-	-
$\boldsymbol{Q}_{uF}(i,i)$	1500.0	$\boldsymbol{Q}_{\Delta\alpha}(i,i)$	50000000.0
$\boldsymbol{Q}_F(i,i)$	1.0	$\boldsymbol{Q}_F(i,i)$	0.01
$\boldsymbol{Q}_u(i,i)$	500.0	$\boldsymbol{Q}_{\Delta F}(i,i)$	1.0

Table 7.4: Environmental forces from irregular sea state

Parameter	**Description**	**Value**
H_s	Significant wave height	1.0 m
T_p	Wave peak period	8 s
N_w	Number of wave components	50 -
γ	Jonswap-spectrum parameter	3 -
T_d	Lower wave spectra period	0.2 s
T_u	Upper wave spectra period	50 s
v_N	North-ward current	-0.1 m/s

7.3.1 Simulation Setup and Tuning

Both the MPC thrust allocation algorithm and the one-step thrust allocation algorithm are tuned to perform as good as possible and to minimize both energy consumption and large oscillations in the commands due to environmental disturbances such as waves, while maintaining stability and robustness. Table 7.3 shows a summary of all the cost function weights. Note that the MPC thrust allocation algorithm is tuned a bit harder, having in general higher costs for the produced thrust and the thrust rates, because it is more robust due to the optimization horizon. Hence, it is expected that the proposed thrust allocation algorithm would have an additional advantages in this benchmarking test.

The environmental forces acting on the vessel in this simulation are the northward current and irregular waves, and the main parameters describing these environmental forces are given in Table 7.4. In the simulation, the vessel is heading north, initially, and is to move 20 m to the north before changing the heading so that it faces east. Then, the vessel is to move 20 m to the east, before changing heading facing south and moving 20 m to the south. Afterwards, the heading is changed to west before the vessel moves 10 m to the west. Lastly, the heading is changed such that the vessel faces northwards before finally moving 10 m to the north.

Note that filters are used to smooth the reference signals before being fed to the DP-controller.

For the proposed algorithm, the integrator time step is set as $\Delta t = 0.005\,s$, the optimization horizon time step is set as $\Delta t_k = 1.0\,s$, and the horizon is set to $T = 30\,s$, resulting in 30 samples in the horizon. The reason for setting the horizon to $30\,s$ is because of the rate limitations for the thruster angles. The thrusters should at least be able to rotate $180\,^\circ$ during the horizon in order to have the possibility to either reverse the thrust or to rotate the thrusters $180\,^\circ$. Hence, with a maximal angular rate of $\pm 10.0\,^\circ\!/s$ the horizon could have been set to T=18.0 s, but because of robustness reasons the horizon is set larger. However, this will be studied in more detail in section 7.4.

The optimization parts in the thrust allocation algorithms are run every Δt_k seconds and only the results from the first sample in the MPC algorithm, $k = 1$, in the horizon are used. The simulation time is set to $4500\,s$, and the DP-control system is initiated at $t = 30\,s$. The simulation results are given in the following.

7.3.2 Simulation Results

Figure 7.6 shows the vessel position and orientation for the two cases. Note that the abbreviation OS is used for the one-step thrust allocation algorithm. As can be seen in the figure, the simulation results show that the vessel position and orientation from the two cases converge, and it is not possible to distinguish the cases from each other. This indicates that both algorithms are equally good at keeping the vessel in position. This can be verified in Figure 7.7 which shows the error between the commanded and the measured positions and orientations. Also in this figure the simulation results seem to converge, except for some small differences. The simulation results also indicate the performance of the DP-controller, which seems to be stable and able to add an appropriate amount of damping to the vessel motion.

Closely related to the outputted thruster commands from the thrust allocation algorithms are the power consumptions from the thruster systems which are shown in Figure 7.8. In the figure, the upper two plots show the power consumption for the two main thrusters, while the third plot shows the power consumption for the bow thruster. The last plot shows a comparison of the total power consumption of the entire propulsion system for the two thrust allocation algorithms.

In the beginning and the end of the simulation, where the vessel is to keep a stationary position and orientation, the power consumptions seem to overlap, having a total power consumption of about $16\,kW$. However, as the figure shows, the power consumptions increase significantly when the vessel is facing east or west. This has to do with the orientation of the ship in comparison to the angle of attack for the

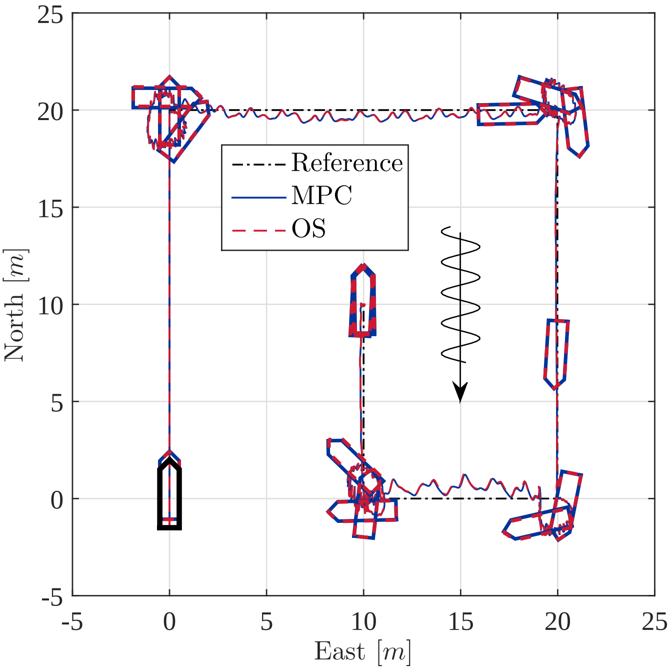


Figure 7.6: North-East plot including heading. The thick black graph in the lower leftmost corner in the figure denotes the initial position and orientation of the vessel

waves and the current forces. Hence, when the vessel is positioned east-westwards, the environmental forces acting on the ship grows significantly since the forces attack the heel of the vessel. From the figure it is also possible to see that the power consumption in this case for the two main thrusters are higher for the one-step algorithm than for the proposed algorithm. For main thruster 2, the maximal power consumption is 88 kW for the one-step algorithm, while only 57 kW for the MPC algorithm. In total, the maximal power consumption for the one-step algorithm is about 414 kW, while 336 kW for the MPC algorithm. Hence, it is not surprising that the one-step algorithm has a higher energy consumption than the MPC algorithm, about 114.6 kWh in comparison to 104.3 kWh for the MPC algorithm. This means that the one-step algorithm consumes about 10 $\%$ more energy than the MPC algorithm in this simulation case. Also, it seems like the oscillations in the power consumption is slightly reduced in the MPC algorithm in comparison to the one-step algorithm, which also can be verified in Figure 7.9 that shows the comparison between the commanded thrust references and the commanded thruster angles for the two main thrusters. As can be seen in the figure, the two algorithms output about the same thrust- and angle commands between the start of the simulations and to about 800 s. After 800 s the vessel starts moving eastwards and both

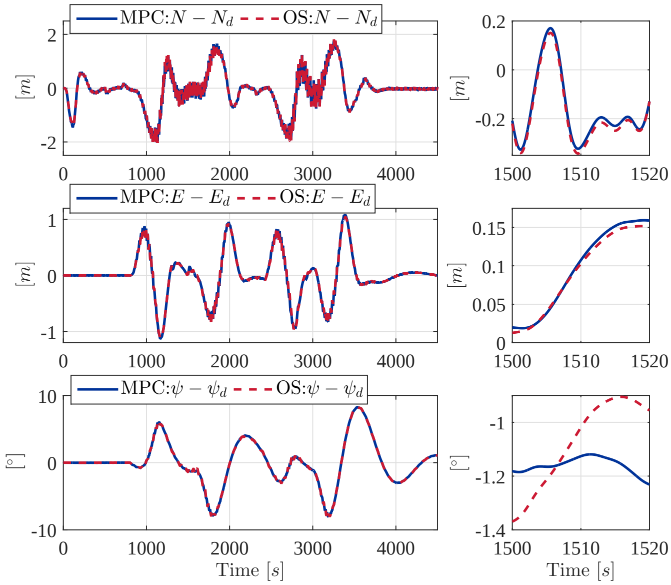


Figure 7.7: Comparison of position and orientation errors for the two thrust allocation algorithms. MPC denotes the MPC-based thrust allocation algorithm while OS denotes the One-Step thrust allocation algorithm

algorithms rotate both the main thrusters counter clockwise, while the produced thrusts are increased significantly when the vessel is oriented in an east-westward direction. This is repeated during the entire manoeuvre. At 3000 s the one-step thrust allocation algorithm decides to rotate main thruster 1 additionally 180 ° and to reverse the corresponding thrust in comparison to main thruster 2 and the main thrusters in the MPC thrust allocation algorithm. However, this is not done in one optimization step, but over a time span of 300 s, which indicates that the resulting commands from the one-step algorithm are not affected by numerical errors due to a low number of allowed iterations, or poor KKT-conditions. On the contrary, it is believed that there exist multiple local optimal minima because of the non-linearities in the one-step problem formulation, and thus, a solving procedure including multiple shooting strategies should be considered if such an algorithm is to be implemented in a realistic manner. However, these results do not affect the total power consumption.

In the simulation time span t =3500 s to the end of the simulation, the two thrust allocation algorithms output about the same commands when taking into consideration that the one-step algorithm has reversed main thruster 1 and that the cor-

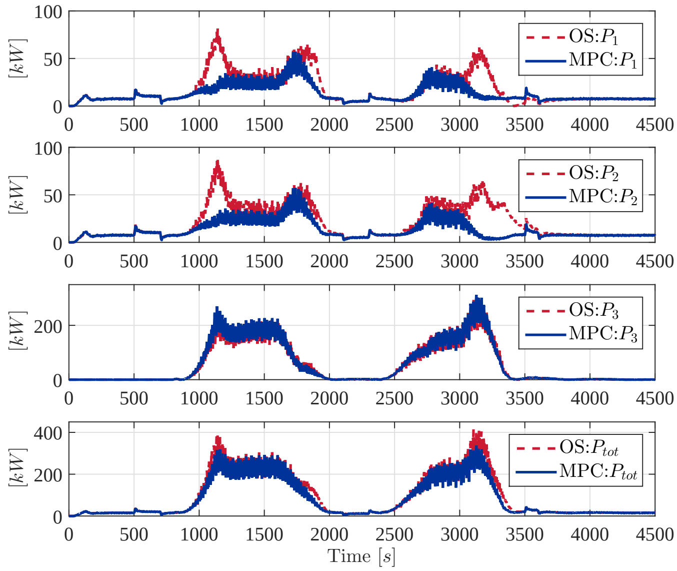


Figure 7.8: Comparison of thruster power and total power consumption of thruster system for the two thrust allocation algorithms

responding thrust command is mirrored. Another interesting observation is that the thrusters in the MPC-case rotates in total 360 ° during the simulation, and thus follows the rotation of the vessel. This illustrates the smoothness of operation of the MPC algorithm. It can also be seen in the figure that both the thruster angle commands and the thrust commands are oscillating less in the MPC algorithm in comparison to the one-step algorithm.

In summary, this benchmarking test has shown that the proposed algorithm has the potential to outperform the one-step algorithm, both when it comes to reduced energy consumption and reduced oscillations in the thrust- and thruster angle commands. When it comes to computational speed, it is of no surprise that the one-step algorithm is faster than the MPC algorithm. The mean computational time for the one-step algorithm in this benchmarking test is 0.385 ms while the mean computational time for the MPC algorithm is 10.56 ms. This means that the one-step algorithm is about 27.4 times faster than the MPC algorithm in this case. However, when having in mind that the horizon of the MPC algorithm is 30 times larger than in the one-step algorithm, the resulting mean computational speed for the MPC algorithm is fast, and since the optimization in the MPC algorithm is initiated only every second, the total algorithm is about 94.7 times faster than real-time. The

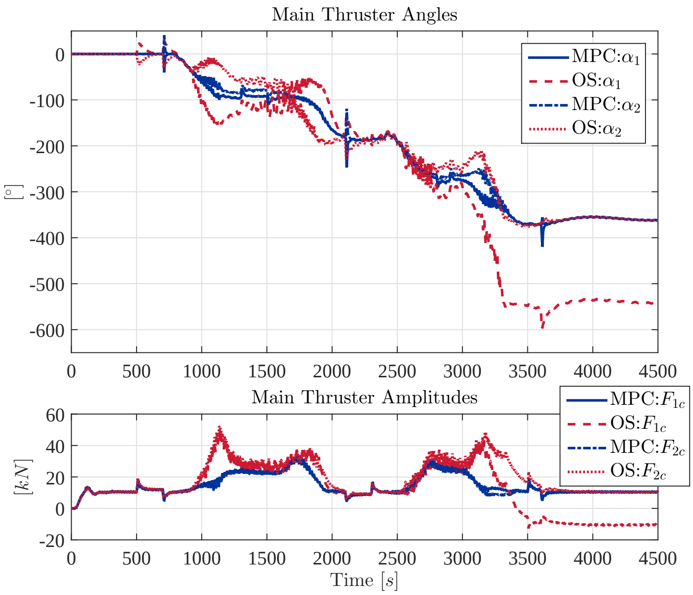


Figure 7.9: Comparison of thruster orientations and thrust amplitudes for the two main thrusters

Table 7.5: Simulation results from benchmarking test

KPI	OS	MPC
Maximal power consumption	$414\,kW$	$336\,kW$
Energy consumption	$114.6\,kWh$	$104.3\,kWh$
Mean computational time	$0.385\,ms$	$10.56\,ms$

main results are summarized in Table 7.5.

In the last part of this chapter the MPC algorithm is to be studied further with respect to built-in filtering properties. Before testing the algorithm with different cost function weights in a realistic simulation case, some preliminary tests are performed in order to map the different properties regarding tuning of cost function weights and the length of optimization horizon.

7.4 Preliminary Case Studies

In the preliminary case studies, a response test is to be applied to the thrust allocation algorithm. The reference signal, $\boldsymbol{x}_d$, from a potential DP-controller, only contains a surge thrust reference, meaning that $Y_d = Mz_d = 0.0$. The surge thrust

Table 7.6: Weighting matrices in different cases. (i, i) denotes the entire diagonal of the matrix

Weight	Case 1	Case 2	Case 3
$\boldsymbol{Q}_k(i, i)$	100.0	100.0	10.0
$\boldsymbol{Q}_T(i, i)$	100.0	100.0	100.0
$\boldsymbol{Q}_{uF}(2, 2)$	10.0	500.0	1500.0
$\boldsymbol{Q}_{uF}(4, 4)$	10.0	500.0	1500.0
$\boldsymbol{Q}_F(i, i)$	0.1	0.1	1.0
$\boldsymbol{Q}_u(i, i)$	10.0	20.0	500.0

reference that is fed to the thrust allocation algorithm consists of a ramp-up and a ramp-down, as well as a step-up and a step-down. Also, the simulation settings and the thrust allocation algorithm time steps are set as in the benchmarking test. The same thruster configuration as used in the benchmarking test, as shown in Figure 7.4, is to be used. The main parameters describing the thruster configuration are the same as listed in Table 7.2, except that now the thruster biasing angle is set to $\pm 20\,°$.

It is expected that the tuning of the cost function weights would have a significant impact on the performance of the proposed algorithm. Hence, as a result of proper tuning, it is expected that the proposed algorithm can be tuned such that thrust rates and thruster angle rates are reduced in order to decrease wearing of the propulsion system. When it comes to optimization horizon lengths, the length should be at least long enough for the algorithm to be able to consider whether it is optimal to rotate a thruster or to reverse the corresponding thrust. In this case it means that the optimization horizon should be at least $T = 18\,s$ because of the angle rate limitations. Hence, it is expected that the main advantage by increasing the horizon even further is gained robustness.

The first preliminary case study treats the cost function weights and is elaborated in the following.

7.4.1 Cost Function Weights

Three different sets of weighting matrices are to be tested and compared for an optimization horizon of $30\,s$. Only changes in four of the weighting matrices are considered, namely $\boldsymbol{Q}_k$, $\boldsymbol{Q}_{uF}$, $\boldsymbol{Q}_F$ and $\boldsymbol{Q}_u$. The weights for the three tuning cases are set as in Table 7.6. Note that case 3 has the same weights as used in the benchmarking test and that all entries in the weighting matrices that are not given in the table are set to zero. The results from the three different cases are compared in Figure 7.10 and 7.11, showing the thrust allocation algorithm output commands

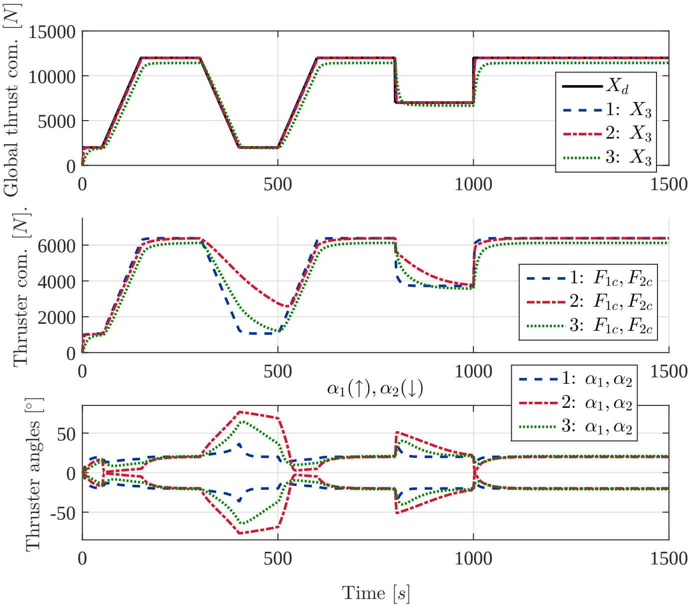


Figure 7.10: Global thrust forces, thruster commands and thruster angle commands for the three different cases. Note that α_1's are plotted in the upper part of the plot (↑), and that α_2's are plotted below (↓)

and the first horizons in the simulations, respectively.

The first plot in Figure 7.10 shows that the three different cases more or less overlap the reference, X_d, except for case 3 which is slightly lower. This has to do with the reduction in the cost function weights $\boldsymbol{Q}_k$ in comparison to the two other cases. However, since a DP-control law with integral effect is used, this will not result in bad performance, which has already been established in the benchmarking test. The second plot in Figure 7.10 shows the characteristics of the optimized thrust commands, F_{1c} and F_{2c}, for the three different cases. As expected, the first case produces the fastest thrust commands, while the second case produces the slowest. The third plot in the figure shows that when the thrust rates decrease, the thruster angles must compensate for that. It is clear from the plot that case two allows more thruster biasing than the two other cases.

The results for the port-side main thruster from the first solved horizon in each case are compared in Figure 7.11. As can be seen in the figure, all the thrust magnitudes from the three cases seem to converge during the time horizon. However, some significant differences between the cases can be seen. The first case seems to focus more on reaching a desired thrust before starting to increase the angle in

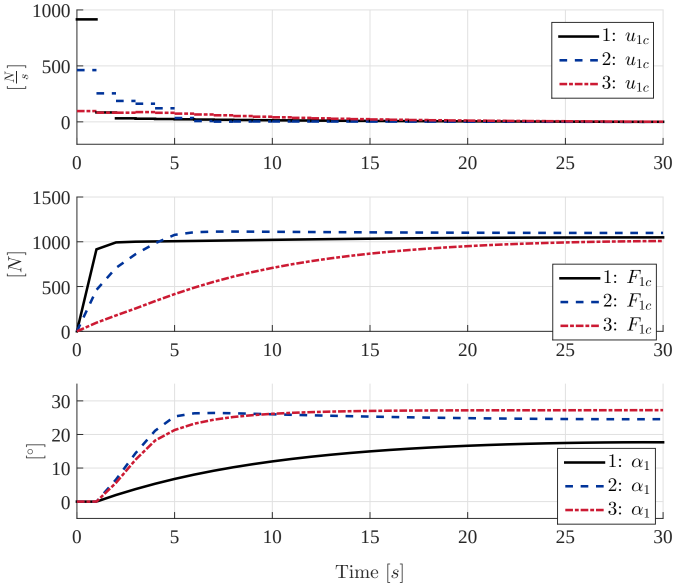


Figure 7.11: Comparison of the first solved time horizon for port-side main thruster for the three cases

comparison to the two other cases. It can also be seen that the thrust rate in the first case is much higher than in the two other cases, causing the thrust magnitude to reach its desired value faster than in the two other cases. When it comes to computational speeds, case 1 is fastest having a mean computational speed of 5.68 ms for each optimization step. Secondly, case 3 had a mean computational speed of 9.08 ms for each optimization step, while case 2 was the slowest one with a speed of 10.4 ms.

The results from this preliminary case study show that by increasing the cost function weights $\boldsymbol{Q}_{uF}$ and $\boldsymbol{Q}_u$ thruster biasing is becoming optimal and reduces the maximal thrust rates. This would be an important property when considering reducing wear of the total propulsion system and the power plant, as well as the amount of consumed energy, which is studied in section 7.5. In the next preliminary case study two different optimization horizon lengths are compared.

7.4.2 Length of Horizon

In section 7.3.1 it was stated that the optimization horizon should be at least 18 s in order to allow the algorithm to determine whether to rotate a thruster or to reverse the corresponding thrust if the thruster could rotate with a maximal angular rate of

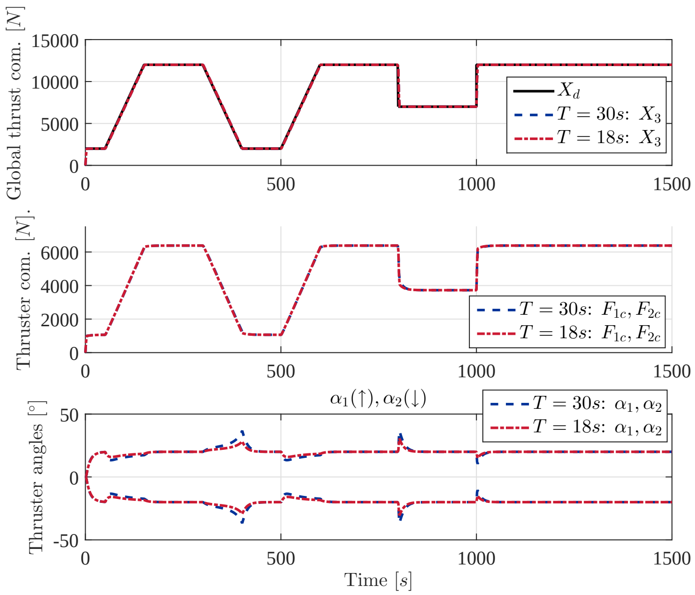


Figure 7.12: Global thrust forces, thruster commands and thruster angle commands for the two different optimization horizon sizes. Note that α_1's are plotted in the upper part of the plot (↑), and that α_2's are plotted below (↓)

10 %/s. However, for robustness reasons, the optimization horizon was set to 30 s. In this preliminary case study, both these two lengths of optimization horizons are tested and compared with the cost function weights for case 1 in Table 7.6. The results from the two different optimization horizons are compared in Figure 7.12 and 7.13, showing the thrust allocation algorithm output commands and the first horizons in the simulations, respectively.

As can be seen in Figure 7.12, both global thrust commands overlaps the thrust reference. Also, in the second plot it is hard to distinguish the thruster commands from the two different optimization horizon lengths. However, some difference can be seen in the thruster angles. The thruster angles in the case with a horizon of 30 s seem to be slightly larger in magnitude in the peaks. Nevertheless, from these results it can be concluded that the two different horizons perform equally in this case.

The results for the port-side main thruster from the first solved horizon in each case are compared in Figure 7.13. As can be seen in the figure, the results show that the two different optimization horizons overlap in the beginning of the simulation, but the case with the shortest horizon converges faster due to the shorter horizon. Also,

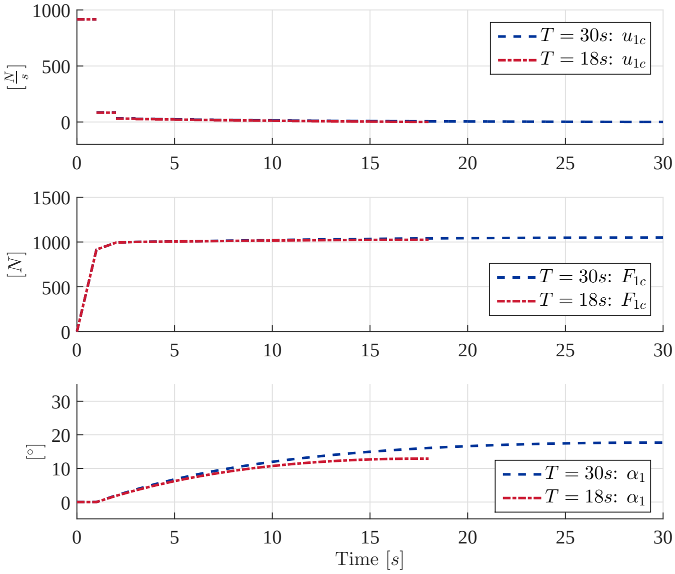


Figure 7.13: Comparison of the first solved time horizon for port-side main thruster for the two different optimization horizon sizes

the thruster angles seem to be slightly larger in the case with the longest horizon, as was also the case in Figure 7.12. When it comes to computational speed, the thrust allocation algorithm with the shortest horizon had a mean computational speed of 1.65 ms for each optimization step, while the other case had a computational speed of 5.68 ms.

In summary, the results show that an optimization horizon of 18 s seems to perform equally good as an optimization horizon of 30 s, mostly due to the maximal allowed angular rates, but also being about 3.4 times faster. However, since the thrust allocation algorithm with an optimization horizon of 30 s is much faster than real-time, this horizon is considered in the rest of the work due to robustness reasons, giving the algorithm even more time to consider rotating or reversing a thruster.

Based on the results obtained in these preliminary case studies, a case study showing the effect of using different cost function weights with respect to reduced oscillations in thruster commands and power consumption in a vessel manoeuvring operation is to be performed.

7.5 Main Case Study

The results from the benchmarking case in section 7.3 and the preliminary case studies in section 7.4 indicate that even though the implementation of the proposed thrust allocation algorithm is fixed, the characteristics of the algorithm can be tailored by tuning the cost function weights. When keeping a vessel in dynamic positioning operations, the wave filter plays an important role in reducing the power consumption, as well as wearing of the power systems, during the operation. This, through filtering out the small wave contributions that keeps the vessel oscillating around its set-position and to catch the drift effects and the slowly varying forces.

Although a wave filter is included and tuned as best, it is impossible to filter out all small oscillatory wave contributions. These contributions are given as input to the DP-controller which tries to compensate for them. This was also the case in the benchmarking test, which showed that even though a wave filter was used, there still was oscillations in the thruster commands, and, hence, the power consumption. However, it is believed that by tuning the thrust allocation algorithm properly, one could obtain an optimal solution with respect to the goal at hand, namely that the thruster angles compensate for these contributions instead of varying the thrust amplitudes. This will be the topic in this case study.

7.5.1 Simulation Setup

The generic offshore vessel model presented in [38] and in Chapter 5 is to be used for testing the three different tuning cases of the thrust allocation algorithm also in this case. The main parameters describing the vessel model and the environmental forces are the same as in the benchmarking test and are given in Table 7.2 and 7.4, respectively.

In these simulations, the vessel is to follow a zig-zag trajectory, as shown in Figure 7.14. Initially, the vessel is to keep its position at $(0, 0)$ in North-East coordinates with the bow facing north, before slowly moving 20 m to the north. Following, the vessel changes heading to face east before moving 20 m eastwards. Afterwards, the vessel changes the heading back to due north before moving additionally 20 m northwards. When the new position has been reached, the vessel changes heading to face west, before moving 40 m westwards. Again, the vessel changes the heading back to due north before moving additionally 20 m to the north. Following, the heading of the vessel is changed to again face east before moving 20 m eastwards. Finally, the vessel changes heading back to due north before moving to the final position 20 m northwards. It is expected that this manoeuvring will stress test the algorithm and the wave filter such that different effects can be reflected in the

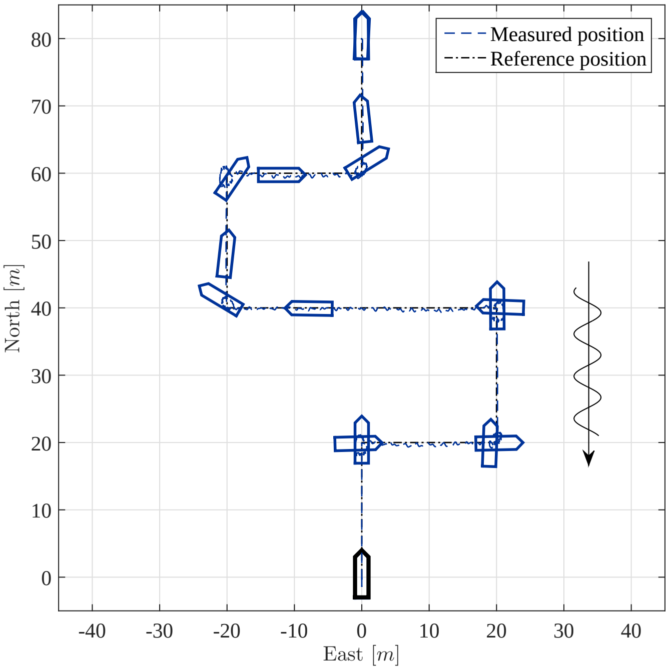


Figure 7.14: North-East plot including heading. The thick black vessel outline denotes the initial position and orientation of the vessel

simulation results as well as ensuring robustness for the proposed algorithm. The simulation results are presented in the following.

7.5.2 Simulation Results

Before presenting the diverging simulation results from the three different tuning cases, the converging simulation results are given. These results include the position and the orientation of the vessel, the global thrust commands compared to the global optimal thrust commands from the thrust allocation algorithm, and the filtered measurements from the wave filter. For the sake of order, the data in the following figures have been obtained from the first tuning case.

Figure 7.14 presents the position and the heading of the vessel in a north-east plot including the heading of the ship for given time steps. As can be seen in the figure, the vessel seems to keep its position and heading quite well in addition to follow the new position commands. These observations are verified by the results shown in Figure 7.15 which compares the measurements $(_m)$ to the commands $(_d)$ and the filtered measurements $(_f)$. The results in this figure also indicate that the wave filter works well in filtering out most of the fastest oscillatory effects from the

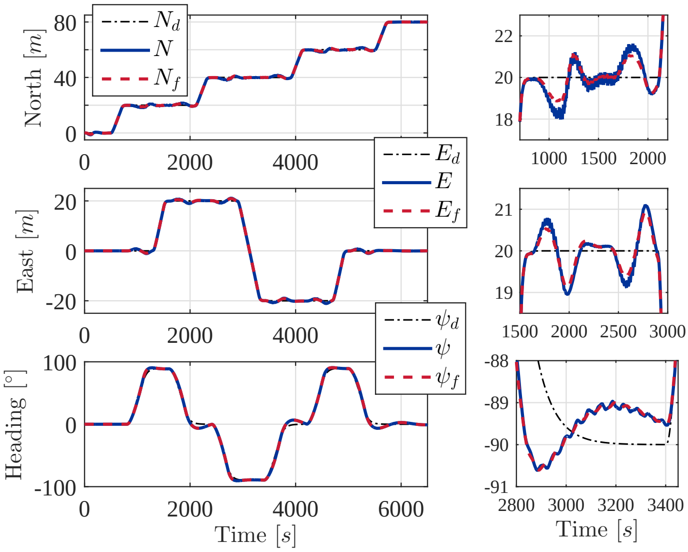


Figure 7.15: Position and orientation of the vessel, both references, measurements and filtered measurements. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

waves, as can be seen in the magnified plots to the right in the figure. The same can be concluded when looking at the filtered position rates and the heading rate compared to the corresponding measurements in Figure 7.16, although there still are some oscillations present in the filtered states. These oscillations will be fed to the DP-controller and cause additionally oscillations in the power consumption if not being suppressed by the thrust allocation algorithm.

In Figure 7.17 the output from the DP-controller is compared with the corresponding output from the thrust allocation algorithm. As can be seen in the figure, the global thrust signals, $\boldsymbol{x}$, converge to the commanded global thrust signals, $\boldsymbol{x}_d$, except for when the commanded rate magnitudes become too high as can be seen in the comparison between Y_d and Y_3. However, the thrust allocation algorithm seems to handle such limitations as it is supposed to.

Even though the commanded global thrust from the DP-controller and the corresponding global thrust from the thrust allocation algorithm overlap in the three cases, the optimal thruster commands and the thruster angle commands are different. Figure 7.18, 7.19 and 7.20 show these results for case 1, case 2 and case 3, as given in Table 7.6, respectively.

For case 1, Figure 7.18 shows that the thruster angles have fast oscillations with

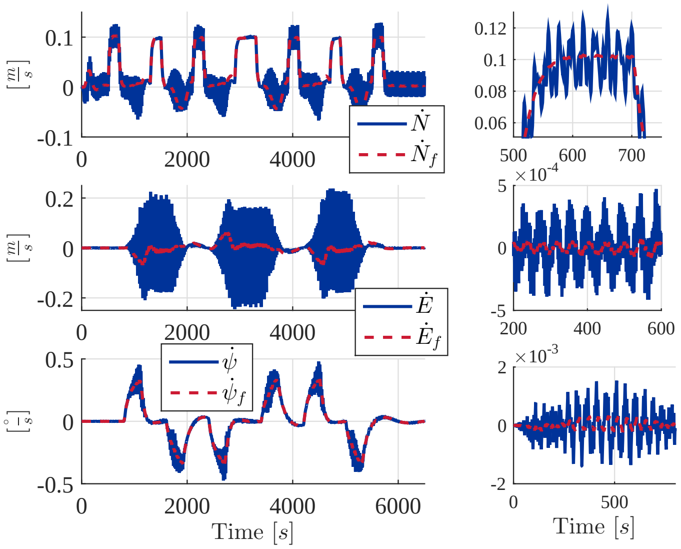


Figure 7.16: Position and orientation rates of the vessel, both measurements and filtered measurements. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

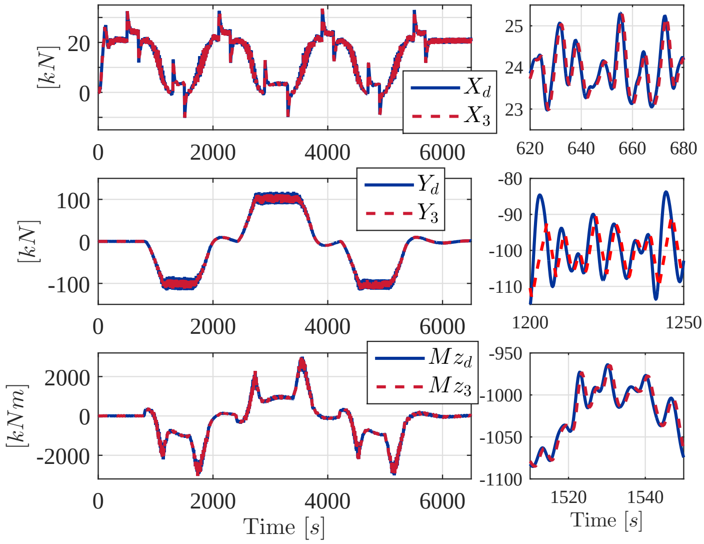


Figure 7.17: DP-commands compared to global optimal thrust from the thrust allocation algorithm. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

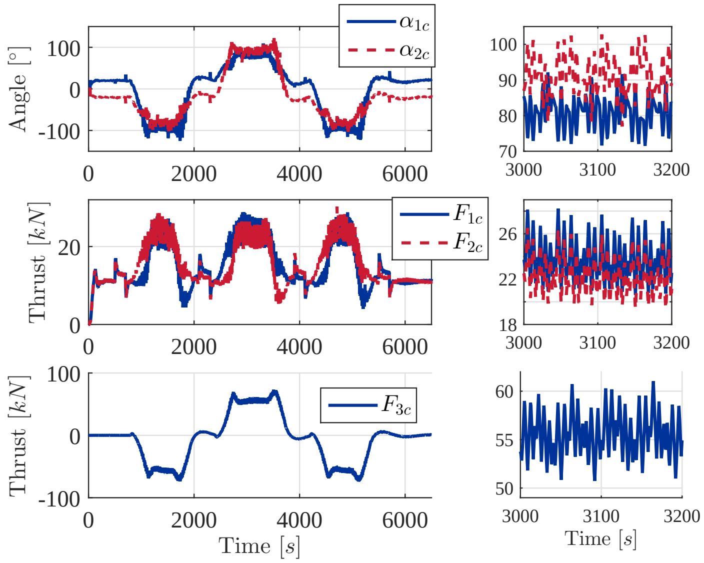


Figure 7.18: Thrust commands and angle commands compared to measured thrusts and angles for case 1. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

small maximal amplitudes of about 5 °, in addition to slower oscillations with an amplitude of about 10 °, when the waves and current encounter the vessel's heel. Also, the main thrusters oscillate with a thrust amplitude of about 3.5 kN as well when the vessel is oriented east-westwards. The thruster angles seem to follow their biasing angles quite good except for when a vessel position change is initiated or when the vessel is parallel with the wave beam. Since the bow thruster is fixed, the thrust command signal seems to oscillate quite a bit as well having an amplitude of about 5 kN. As will be seen later on, the results from the bow thruster can also be improved by tuning the thrust allocation algorithm properly, even though there is no controllable thruster angle to work with.

Figure 7.19 shows the corresponding results from case 2. As can be seen in the figure, the fastest oscillations in the thruster references are decreased to about 2.5 kN in magnitude, a reduction of about 30 $\%$ compared to the results from case 1. However, the amplitude of the oscillations in the thruster angles are quite different in this case in comparison to case 1. In the beginning of the simulation the amplitude is increased from about 1 ° to about 7.5 ° when compared to the first case. When the vessel is oriented east-westwards the picture is different. The amplitude of the fastest oscillations in the thruster angles are decreased to about 4 ° while the amplitude of the slowest oscillations are increased to about 15 °. It is also possible to

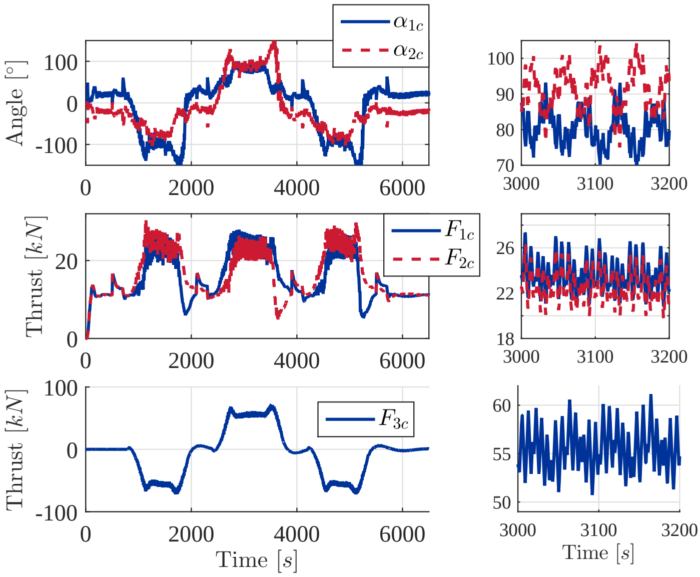


Figure 7.19: Thrust commands and angle commands compared to measured thrusts and angles for case 2. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

see that thruster biasing is more accepted in this cases when looking at the peak in the thruster angle command α_2 around $t = 3500\,s$. The figure also shows that the fastest oscillations in the thruster commands for the bow thruster are reduced to about 3.5 kN.

The results from case 3 are given in Figure 7.20. The results show that the oscillations in the thrust are decreased even more, to about 1 kN in amplitude, a reduction of about 70 $\%$ in comparison to the results from case 1. Also the oscillations in the thrust angles are decreased in the entire simulation, to about 1 ° in the beginning of the simulation as well as for the fastest oscillations when the vessel is oriented east-westwards, while the maximal amplitude of the slowest oscillations is decreased to about 2 °. The reason for this is that the weights in $\boldsymbol{Q}_{uF}$ and $\boldsymbol{Q}_u$ are increased significantly in comparison to the error weights in $\boldsymbol{Q}_k$ and $\boldsymbol{Q}_T$. This means that the thrust allocation algorithm allows more error than in the two previous cases. However, the errors that are allowed are only due to the small oscillations, meaning that the chosen weight combination in the thrust allocation algorithm works more or less as a low pass filter without phase differences. This error is completely fine since the integrator in the DP-controller makes sure that the vessel does not experience a drift-off due to this error.

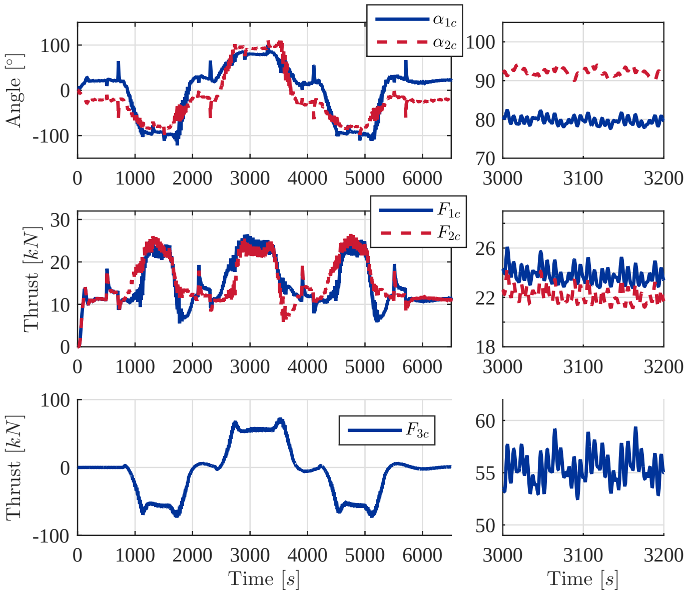


Figure 7.20: Thrust commands and angle commands compared to measured thrusts and angles for case 3. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

When it comes to the fastest oscillations in the bow thruster commands, the maximal amplitude is reduced even more to about $2\,kN$, which lead to a smoother power consumption, as can be seen in Figure 7.21. In case 1 the total power consumption was about $200.9\,kWh$, and the increase of consumed power in case 2 and 3 were $1.3\,\%$ and $0.1\,\%$, respectively. Also, it can be seen that the noise in consumed power is significantly reduced in the third case in comparison to the two others. This means that the extra fuel cost for tuning the thrust allocation algorithm to filter out environmental disturbances that have not been suppressed by the wave filter is negligible.

The results regarding the filtering properties for the thrust allocation algorithm are summarized in Figure 7.22, showing a comparison of the power spectral density of the commanded thrust force and orientation of thruster 1. As the figure indicates, the fastest thrust oscillations are largest in simulation case 1 while the fastest thruster angle oscillations are largest in simulation case 2. However, the slowest oscillations in the thrust commands are largest in case 3 while the slowest thruster angle oscillations are largest in case 2. This means that case 1 and case 3 is better at compensating for the slowly varying environmental disturbances while case 2

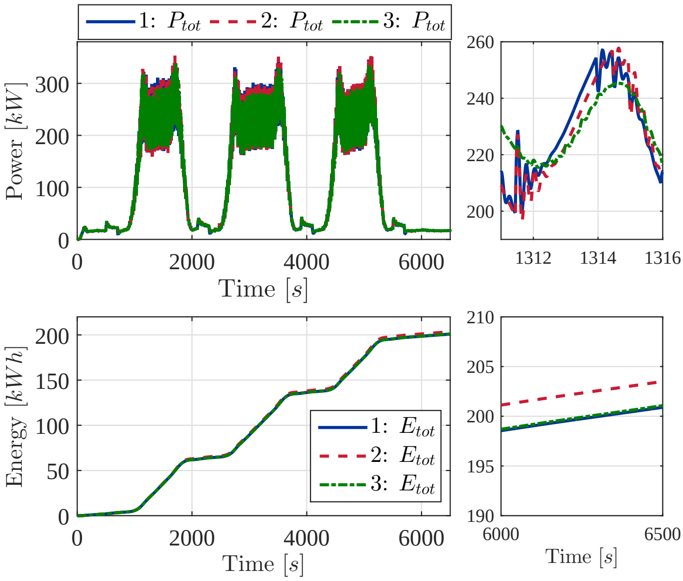


Figure 7.21: Comparison of power consumption and total energy consumption for the three simulation cases. Note that the small plots to the right in the figure shows a magnified region of the plots to the left

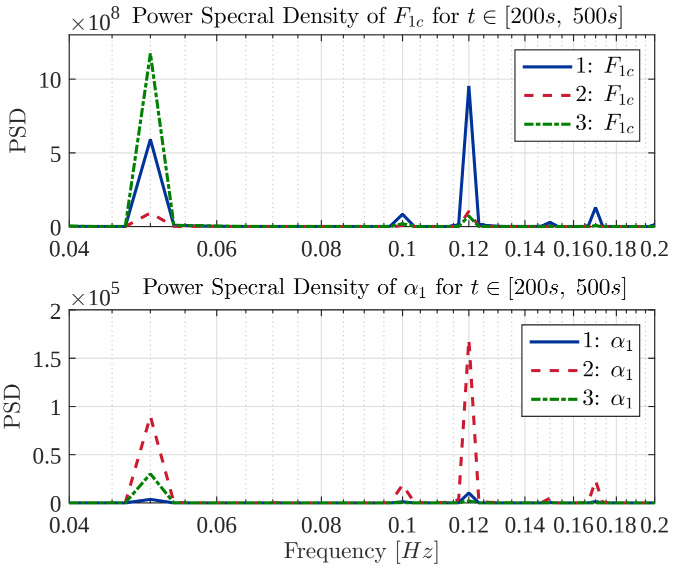


Figure 7.22: Power spectral density of commanded thrust force and orientation for main thruster 1

filter out as much as possible by using the thruster angles. In summary these results show that the set of cost function weights used in case 3 would be the most preferred ones since it reduces the fastest oscillations in the thruster commands as well as in the power consumption, while being able to compensate for slowly varying environmental disturbances. This, in addition to keeping the total energy consumption at a minimum. The simulation results from this case study, as well as from the benchmarking test and the preliminary case studies, also argue for the use of thruster biasing in order to reduce wearing of the systems, even though it increases the total energy consumption.

The mean computational speeds in these simulations are given as 9.06 ms, 16.37 ms and 15.60 ms for case 1, 2 and 3, respectively. Even though case 2 is the slowest one, it is still about 61 times faster than real-time.

7.6 Conclusion

In this chapter a proposed non-angular MPC-based thrust allocation algorithm has been presented and tested. The reason for formulating the optimization problem in thrust vectors instead of thrust amplitudes and angles, was to reduce non-linearities in the thrust allocation problem in order to make the thrust allocation algorithm fast solvable, at least in real-time, which was accomplished.

The proposed thrust allocation algorithm was benchmarked against a commonly used one-step thrust allocation algorithm, and the results showed that the proposed thrust allocation algorithm outperforms the one-step algorithm, both when it comes to total energy consumption as well as reducing oscillations in the thrust commands and thruster angle commands. Even though the one-step algorithm was 24.7 times faster than the MPC algorithm, the MPC algorithm had a real-time index (RTI) of about 94.7 in the benchmarking case. The benchmarking test also showed that the proposed thrust allocation algorithm worked smoothly when the commanded thruster angles increased, which indicates that the calculation procedures in Algorithm 7.2 and 7.3 work properly.

Three sets of rather coarse cost function weights were chosen in order to test the thrust allocation algorithm and to show different optimization strategies in the first preliminary case study. The sets of weights were then tested on a vessel model in order to see if the differences in weights affected the overall goal of keeping the vessel in position while reducing thrust oscillations, due to fast wave effects that could not be filtered out with the wave filter, by using thruster biasing and actively compensate for these fast oscillations with the thruster angles. The results showed that even though the weighting matrices were different, the position and orientation of the vessel were maintained. Also, the tuning of the proposed algorithm affected the total energy consumption of the propulsion system. It would be interesting to include a power plant in the simulation, as the one presented in Chapter 6, to see how the tuning of the thrust allocation algorithm affects the overall system. This will be done in Chapter 8. Also, it should be mentioned that in the case study there are situations where the thruster angles approach orientations of $\pm 90\,^{\circ}$. In reality, such orientations would cause hydrodynamic thruster-thruster interactions, decreasing the efficiency of the main thruster affected by the wake of the other main thruster. However, such effects are not included in the vessel model, nor have restrictions for this been included in the thrust allocation algorithm, although it has been mentioned. This is left out for future work.

Even though the energy consumption increased negligibly from case 1 to case 3, the results also showed that thrust oscillations and thrust rates were decreased significantly from case 1 to 3. This points to the fact that the thrust allocation algorithm can be tuned such that wearing of the propulsion system and power plant can be reduced due to unfiltered environmental effects in the DP-controller commands.

Part III

Applications of Co-Simulations

The science of today is the technology of tomorrow
- Edward Teller

CHAPTER 8

Maritime Applications of Co-Simulations

This chapter is based on [42, see **P10** in section 1.6] where different applications of distributed co-simulations in the maritime industry are discussed. Four different case studies are presented in this chapter where the first one illustrates how co-simulations can be used for improving collaboration between researchers as the platform for connecting different work together into more complex simulators [43, see **P11** in section 1.6]. The second case study illustrates how co-simulations can be used for optimizing system integration on a higher level, such as optimizing the interplay between the power plant in a marine vessel, the propulsion system and high level control systems such as a DP-control system. The third case study illustrates how hardware can be included into the co-simulation loop [41, see **P9** in section 1.6]. The last case study illustrates how co-simulations can be used as an effective design tool for testing different vessel- and equipment configurations in different scenarios in high fidelity simulations in a fast and generic manner.

8.1 Introduction

The ship design process has traditionally been characterized by customized solutions, strict time and resource constraints and much of the major design variables were taken upfront based on experience rather than a scientific first principles based engineering approach. However, for most of the engineering disciplines involving scientific knowledge, methods and tools are now highly developed and adopted, but with a component or subsystem focus rather than a complete ship system design focus. Now there is a development in ship design towards higher degree of verification needed, especially for special- and high-technology vessels such as autonomous ships. One example of such an verification is the DP capability of an offshore vessel in complex operations. This is discussed in [200] where the importance of analysing the dynamical DP capability of marine vessels using suited dynamical models is highlighted. How better to support the design process at different levels and detail, to seek optimized system solutions, and at the same time being able to validate the actual designs is the vision for a future ship design framework of methods and tools. Distributed co-simulation technology is one building block for reaching this vision.

Research and development activities in the maritime industry are characterized by specialists working from different angles on joint projects using specialized computer software to optimize designs before any prototypes are built. Since the costs of building a prototype are significant, often only a single one is scheduled, if one is even considered at all. This is especially the case when designing a new ship where the prototype is in fact the ship delivered to the customer at the end of the project. Despite the fact that the number of prototypes are significantly reduced, the iterative process of obtaining the best design remains more or less the same [201], except that the iterations have moved from the workshop floors to the engineering offices and into suited computer software, mostly as mathematical models designed to replicate the dynamical characteristics of the physical design. This increases the expectations and sets high requirements for the specialists as well as the software, trying to realize the product properties specified by the customer in the short lifetime of the project. On the other hand, the project manager expects the project group to oblige the customer and deliver a satisfying product within the time agreed upon in order to obtain a financial surplus rather than large financial penalties and a dissatisfied customer.

When the iterative design process is automated on a computer and where experience is mixed with a scientific first principles based engineering approach, it may be difficult to quantify a *good ship* for its purpose. Now, not only experience are used for evaluating the design, but also new data obtained from the iterative process based on mathematical models of the vessel and its subsystems. Different key performance indicators (KPIs) can be used, but these need to be defined and able to proclaim a good vessel design. The major question then becomes what is a better ship, and has been discussed in [202]. However, this question will not be treated in any detail in this chapter, nor the iterative design process.

In the previous chapters the main focus has been devoted to background theory and different building blocks which are to be put together in this chapter. While Part I, including the chapters 2-4, focused on co-simulation theory, the ViProMa project and related challenges, such as stability and accuracy in co-simulations as well as how tightly coupled systems in co-simulations can be solved, Part II, including the chapters 5-7, presented some generic domain models and control systems. The marine offshore vessel model presented in Chapter 5 includes many different subsystems e.g. a wave-filter, DP-control system and a propulsion system. These subsystems can be implemented as separate subsystems in co-simulations and to be used in a modular fashion. Another important subsystem in a marine vessel is the vessel's power plant and a generic power plant model including hybrid causality generator models was presented in Chapter 6. By combining the different submodels presented in Chapter 5, the power plant model in Chapter 6, the MPC-based thrust allocation algorithm presented in Chapter 7 and more sophisticated propulsor drive models in a co-simulation, one is able to simulate realistic dynamical interactions between equipment due to predefined operational conditions and environmental disturbances, as well as investigating the characteristics and performance of the vessel. Applications of such investigations will be the main focus in this chapter.

During the ViProMa project period four different use-cases and demonstrators were made mainly for research purposes and for testing the open source co-simulation master algorithm Coral, see section 2.4.1, and will be presented in this chapter. These case studies show a wide range of the use of co-simulations in maritime industrial applications, as well as in research projects, and include Hardware-In-the-Loop (HIL) in co-simulations, collaboration between researchers using co-simulations, optimizing system integration using co-simulation and testing different vessel configurations using co-simulation. Note that the main focus in these case studies is not the models or the simulation results themselves, but the applicability and advantages of utilizing co-simulation technology in complex engineering tasks in the maritime industry, although the simulation results also have research value in themselves. Section 8.3 presents the open source vessel simulator, as first mentioned in section 2.4.2, and will be given much attention.

8.2 Research Collaboration

When studying complex dynamical systems that grow large because of many high fidelity subsystems, different specialized software are hard to combine in a generic fashion. However, by utilizing distributed co-simulations, researchers can work on different subsystems in their preferred software without being concerned about compatibility except for model interfaces.

In [43] five researchers looked into using a shaft generator to reduce the transients of a two stroke maritime engine powering a very large crude oil carrier (VLCC) in a transit operation affected by significant wave loads. In such operations the propeller might ventilate causing varying loads on the propeller and, hence, the propulsion system. While two of the researchers were researching wave loads and ventilation of propellers, the three other researchers were looking into the power systems.

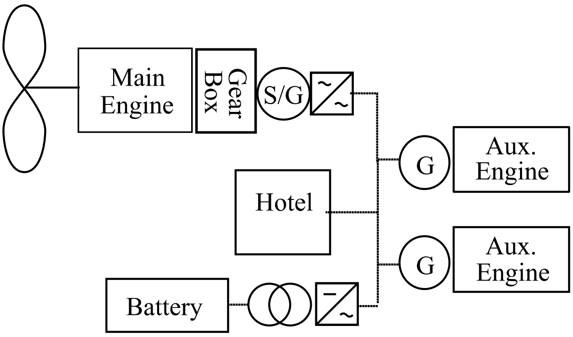


Figure 8.1: Total system overview of a vessel's propulsion system and power plant. The figure is obtained from [43]

8.2.1 System Overview

The total power system, including a propeller, a main engine, a gear box, a shaft generator, two gensets, a hotel load, a battery system, a converter and a rectifier, is illustrated in Fig. 8.1. Note that two gensets and the hotel load are as modeled in Chapter 6, having the same parameters. A short introduction to the subsystems used in the co-simulation are given in the following in order to illustrate the complexity of the total system, and the reader is referred to [43] and the references given therein for detail matters.

Main Engine, Gear Box and Shaft System

In order to capture the full system dynamics of the diesel engine system, all the essential components were included in the engine model. These components include turbochargers, scavenge air coolers, pipe components, scavenge air reciever volume, individual cylinders, an exhaust gas receiver and a wastegate valve. A schematic overview of the connected components is given in Figure 8.2. A detailed description of these components are out of scope here, but are thoroughly elaborated in [203, 204]. The main parameters describing the two-stroke diesel engine are given in Table 8.1.

The gear box between the main engine and the shaft generator in Figure 8.1 is assumed ideal, meaning that it has no friction, and has a constant gear ratio of 8:1. The shaft system connecting the propeller, the shaft generator and the main engine is given as

$$J_T\dot{\omega} + f(\omega) = \tau_T \tag{8.1}$$

where $J_T = 323000\,\mathrm{kgm}^2$ is the total inertia of the propulsion system including the main engine crank shaft, the main shaft, the propeller and the gear box, ω is the shaft speed, $f(\omega)$ is a second order friction function polynomial and τ_T is the

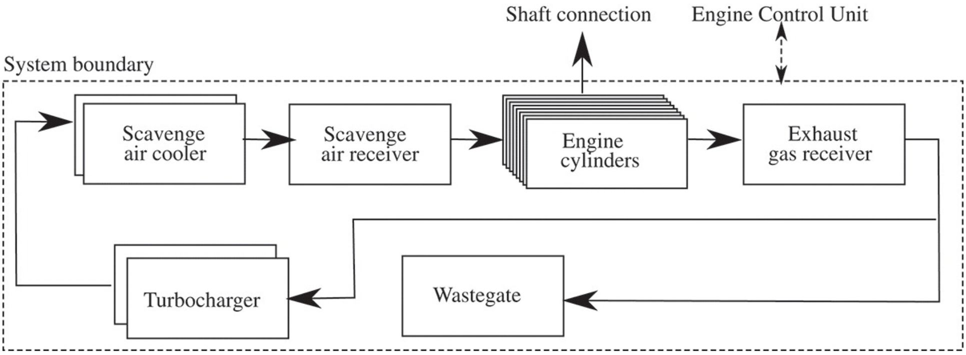


Figure 8.2: Overview of the main engine model [203]

Table 8.1: Particulars of the main diesel engine

Type	Wartsila 8RT-flex68D
Bore	680 mm
Rated MCR	25040 kW
Speed at rated power	95 RPM
Stroke	2720 mm
Mean effective pressure	20 bar
Number of cylinders	8 -
Turbocharger	2×ABB A175-L35

resulting shaft torque including the propeller torque, the main engine torque and the shaft generator torque.

Propeller and Hull Model

The propeller model is based on results from frictional drag calculations obtained from the open-source program *Openprop*, that utilize vortex lattice lifting line theory, in addition to the program *Javafoil* [205]. These results were compared and evaluated against available experimental open water data, and full-scale open water curves were obtained for the propeller design to calculate thrust and torque at given propeller speeds and vessel speeds. In order to consider the effect of waves on the propulsion system, thrust and torque losses due to propeller emergence, free surface effects and the Wagner effect have been modeled as suggested in [206, 207]. These effects have been considered in quasi-steady sense as propeller depth varies much slower than the propeller's rate of rotation. The resulting thrust and torque are assumed to vary with wave frequency and higher order harmonics have been neglected. Details regarding the propeller model can be found in [43, 208] and the main propeller parameters are given in Table 8.2.

Table 8.2: Propeller geometry

Diameter	9.86 m
No. of blades	4 -
Hub diameter	1.53 m
Projected blade area ratio (A_e/A_0)	0.431 -
Pitch/Diameter gradient (mean)	0.47 -
Skew	21.15 °
Rake	0 °

The hull model, in contrast to the hull model derived in Chapter 5, is a 1 degree of freedom (DOF) model of the virtual ship KVLCC2 [209]. The ship motion response amplitude operators (RAOs), such as the surge-, pitch- and relative stern motion were calculated using linear strip theory, utilizing potential theory and pressure integration, implemented in the ShipX Veres software developed by SINTEF Ocean. From the motion responses of the hull, added resistance coefficients were also computed in irregular waves for different peak frequencies and wave directions using the Pierson Moskowitz wave spectrum [142].

In summary, the dynamics of the KVLCC2 vessel can be expressed as

$$(m + m_A)\ddot{x} = (1 - t_d)T - (\frac{1}{2}\rho S C_T \dot{x}^2 + R_1) \tag{8.2}$$

where m is the mass of the ship, m_A is the added mass in surge, x is the surge displacement, T is the thrust produced by the propeller, t_d is the thrust deduction factor, ρ is the density of seawater, S is the wetted surface area of the ship, C_T is the resistance coefficient of the ship modeled as a function of ship speed and R_1 is the sum of 0th and 1st order added resistances in waves. Due to lack of knowledge of the variation of the thrust deduction coefficient in waves, it has been assumed constant. However, note that R_1 and T are time dependent inputs. The specific hull geometry details are found in Table 8.3.

Shaft Generator and Converter

The shaft generator is modeled as an asynchronous electrical motor in the $(d, q, 0)$-reference frame and according to [164] the dynamics are given as

$$\begin{aligned} \dot{\boldsymbol{\psi}} &= \boldsymbol{K}(\omega_s, \omega_r)\boldsymbol{\psi} + \boldsymbol{R}\boldsymbol{i} + \boldsymbol{E}\boldsymbol{u}_s \\ \boldsymbol{i} &= \boldsymbol{L}^{-1}\boldsymbol{\psi} \end{aligned} \tag{8.3}$$

where $\boldsymbol{\psi} = [\psi_{ds}, \psi_{qs}, \psi_{dr}, \psi_{qr}]^\top$ is the magnetic flux linkage vector, $\boldsymbol{i} = [i_{ds}, i_{qs}, i_{dr}, i_{qr}]^\top$ is the current vector, $\boldsymbol{u}_s = [u_{ds}, u_{qs}]^\top$ is the dq stator

Table 8.3: Hull design parameters

Length between perpendiculars	320.0 m
Length at waterline	325.5 m
Breadth at waterline	58.0 m
Depth	30.0 m
Draft	20.8 m
Displacement	312622 m^3
Block coefficient (C_B)	0.8098 -
Design Speed	7.97 m/s

voltage vector given as input by the converter, $\boldsymbol{i}_s = [i_{ds}, i_{qs}]^\top$ is the dq stator current vector given in feedback to the converter and

$$\boldsymbol{K}(\omega_s, \omega_r) = \begin{bmatrix} 0 & n_p\omega_s & 0 & 0 \\ -n_p\omega_s & 0 & 0 & 0 \\ 0 & 0 & 0 & n_p(\omega_s - \omega_r) \\ 0 & 0 & -n_p(\omega_s - \omega_r) & 0 \end{bmatrix}$$
$$\boldsymbol{E} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \boldsymbol{R} = \begin{bmatrix} -r_s & 0 & 0 & 0 \\ 0 & -r_s & 0 & 0 \\ 0 & 0 & -r_r & 0 \\ 0 & 0 & 0 & -r_r \end{bmatrix} \tag{8.4}$$
$$\boldsymbol{L} = \begin{bmatrix} L_s & 0 & L_m & 0 \\ 0 & L_s & 0 & L_m \\ L_m & 0 & L_r & 0 \\ 0 & L_m & 0 & L_r \end{bmatrix}$$

where n_p is the number of pole pairs,

$$\omega_s = \frac{i_{qs} L_m r_r}{L_r \psi_r + \epsilon} + \omega_r n_p, \tag{8.5}$$

where ω_s is the synchronous speed,

$$\psi_r = \sqrt{\psi_{dr}^2 + \psi_{qr}^2} \tag{8.6}$$

is the rotor flux, ω_r is the rotor speed, r_s is the stator resistance, r_r is the rotor resistance L_s is the self inductance for the stator, L_r is the self inductance for the rotor, L_m is the motor inductance and ϵ is a small number added in order to avoid dividing by zero when $\omega_r = 0$. Note that the electromagnetic torque is given similarly as in (6.6),

$$T_e = (\psi_{qr} i_{dr} - \psi_{dr} i_{qr}) n_p \tag{8.7}$$

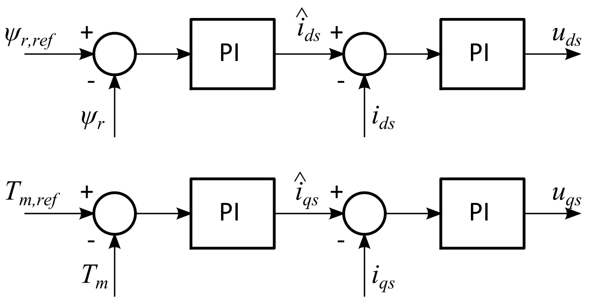


Figure 8.3: FOC structure for controlling the shaft generator

The converter is assumed ideal, which means that the voltages fed to the electrical motor are given by the motor controller and the current given in feedback to the power grid is calculated as

$$\boldsymbol{i}_s = \frac{1}{||\tilde{\boldsymbol{u}}_s||_2^2 + \epsilon} \begin{bmatrix} P\tilde{u}_{ds} + Q\tilde{u}_{qs} \\ P\tilde{u}_{qs} - Q\tilde{u}_{ds} \end{bmatrix} \tag{8.8}$$

where $\tilde{\boldsymbol{u}}_s$, $\tilde{u}_{ds}$ and $\tilde{u}_{qs}$ are the low-pass filtered voltages $\boldsymbol{u}_s$, u_{ds} and u_{qs}, respectively, in order to avoid algebraic loops, $\boldsymbol{u}_s$ is the voltage vector provided by the power plant, u_{ds} and u_{qs} are calculated by the motor controller and

$$\begin{aligned} P &= \frac{u_{ds}i_{ds} + u_{qs}i_{qs}}{\eta} \\ Q &= \sqrt{\left(\frac{P}{PF}\right)^2 - P^2} \end{aligned} \tag{8.9}$$

where η is the efficiency of the converter and PF is the power factor, both assumed constant.

The shaft generator is assumed controlled by a field oriented controller (FOC), such as in [210], where the torque of the motor is controlled by the torque component of the stator current while maintaining a constant rotor flux, as described in [154], and the control structure can be as illustrated in Figure 8.3 consisting of four PI-controllers. In the figure, $\psi_{r,ref}$ and $T_{m,ref}$ is the reference rotor flux and motor torque, respectively, T_m is the measured or estimated motor torque and $\hat{i}_{ds}$ and $\hat{i}_{qs}$ are the current references for the second PI control laws. The main parameters for the shaft generator are listed in Table 8.4 and are the same as used in [164].

Battery System and Overall Control

The battery system is added to the system in order to absorb/provide power from/to the shaft generator in order to not overload the power plant. The battery system has

Table 8.4: Shaft generator parameters

Rated output	3300 kW	Inertia rotor	175 kgm^2
Voltage	690 V	L_s	0.00156 H
Frequency	60.3 Hz	L_r	0.00155 H
Speed	1199 RPM	L_m	0.00151 H
Current	3270 A	r_s	0.00195 Ω
Poles	6 -	r_r	0.00166 Ω

Table 8.5: Design parameters for battery pack

Energy capacity	1000 kWh
Maximum discharging current	5.6 kA
Maximum charging current	5.6 kA
Nominal Voltage	360 V

been designed such that large amplitudes of the state of charge (SoC) cycles are avoided. The main design parameters for the battery system are given in Table 8.5. The battery system has an interface with the AC power system, through a rectifier and a transformer, as shown in Figure 8.4. A simple capacitance-resistance model of the battery is used since the main aim with this study is the overall system dynamics, not to analyse the battery dynamics. Hence, the battery is also assumed ideally controlled, which means that the battery provides the commanded current with some time delay expressed through a first order transfer function.

Since the idea of including the shaft generator is to reduce the transients of the two-stroke main diesel engine caused by wave loads, a proper control system is needed such that the shaft generator filters the oscillating shaft speed, caused by transient effects on the propeller, by reducing or increasing the shaft speed, and where the battery is used for short time energy storage. An overview of the total control system, including both control of the main engine, the shaft generator and

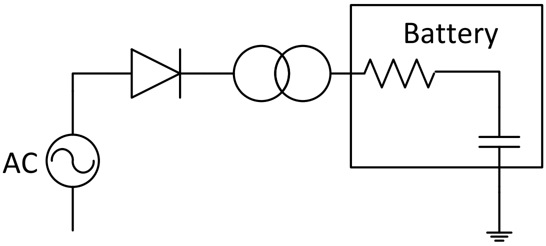


Figure 8.4: Equivalent electrical circuit of the battery system. Figure obtained from [43]

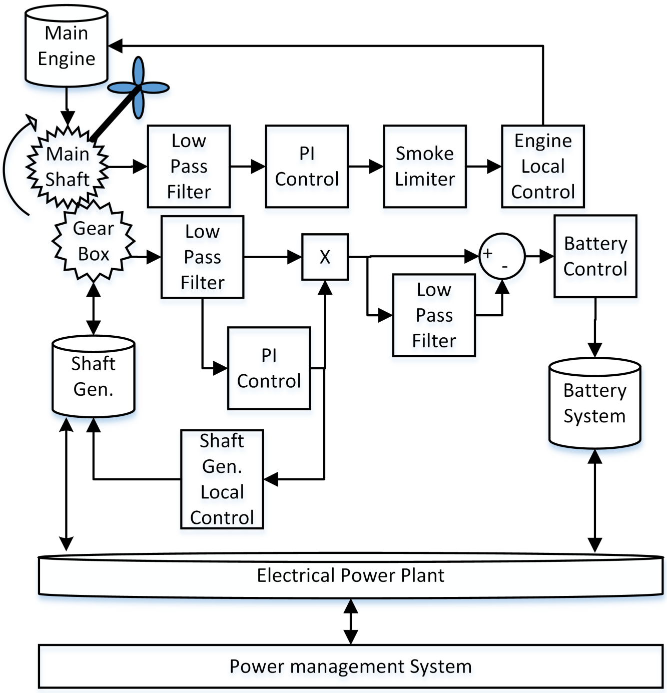


Figure 8.5: Sketch of the control system structure. Figure obtained from [43]

battery system, is shown in Figure 8.5.

As can be seen in the figure, a feed-forward control strategy is used where a prediction of the power used or consumed is made by multiplying the measured speed and the commanded torque. Then, a low-pass filter is used to extract only the power oscillations which are given as feed-forward to the battery system. The entire idea of using a shaft generator to reduce the transients of the main engine is only possible if the shaft generator reacts faster than the two-stroke diesel engine, which is the case here. More details about the battery system and the control structure are given in [43].

8.2.2 Simulation Setup

The power plant, as presented in Chapter 6, including the auxiliary engines, the generators and the hotel load was exported as one FMU, while the two-stroke diesel engine model, the vessel model, the propeller model, the shaft model, the shaft generator model and the battery power pack model were each exported as separate FMUs.

The total simulation model were mainly constructed in the software 20-Sim and

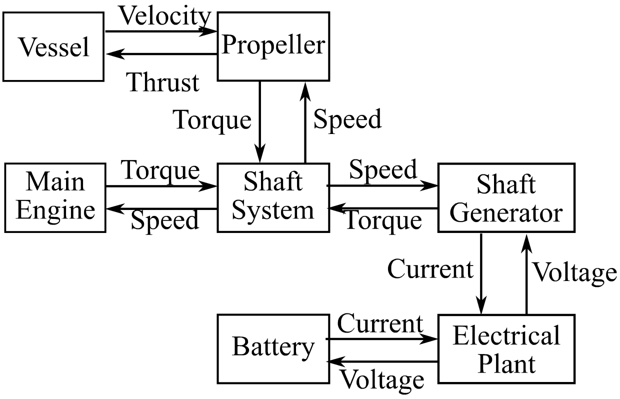


Figure 8.6: Simulation setup. Note that each block represents a FMU in the total co-simulation. The figure is obtained from [43]

Simulink, and simulated as a co-simulation using Coral with the model connections as shown in Figure 8.6. Note that all connections between submodels shown in the figure are power bonds, according to the bond graph modeling theory as presented in section 1.3.2. Because of the amount of computational power needed to solve the total system, and the fact that different modeling software were used to make the dynamical models, performing such a simulation study in a traditional manner as a non-distributed system would have been quite time consuming, both because all systems must be modeled in the same simulation software and because all systems need to be solved using the same numerical solver, despite having quite different dynamical properties.

In [43], 45 different co-simulation cases are performed and a thorough study of how the shaft generator affects both the transients in the two-stroke diesel engine and the fuel consumption is given. Here, only one case is presented, namely a transit case where the VLCC faces head sea with a wave amplitude of 4 m having a wave length of 352 m. The hotel load in the power plant model is in the simulation set to 1000 kW, the total system simulation time is set to 300 s, the global communication time step is set to 10 ms and the different local numerical solvers and respective time steps for each of the subsystems are shown in Table 8.6.

8.2.3 Simulation Results

The simulation results are compared to a conventional propulsion system, e.g. the two stroke diesel engine powering the propeller without any shaft generator. The shaft speed simulation results are shown in Figure 8.7. Note that the conventional propulsion system was also simulated as a co-simulation and that the shaft gener-

Table 8.6: Subsystems and integration methods

Subsystem	Integration method	Time step
Hull	Euler, 1st order	100 ms
Propeller	Euler, 1st order	10 ms
Shaft system	Euler, 1st order	10 ms
Diesel engine	Runge-Kutta, 4sth order	2 ms
Power Plant	Euler, 1st order	0.1 ms
Shaft generator	Runge-Kutta, 4th order	0.05 ms
Battery system	Euler, 1st order	10 ms
Control system	Euler, 1st order	10 ms

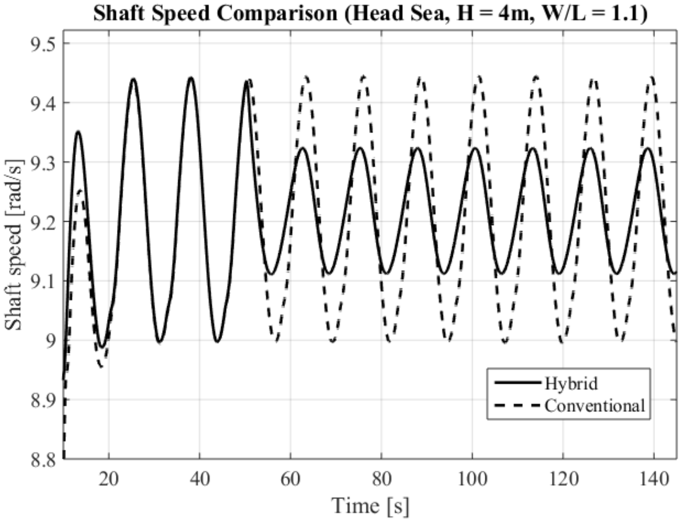


Figure 8.7: Comparison of shaft speeds between the hybrid propulsion system and the conventional propulsion system. The simulation results are obtained from [43]

ator in the hybrid propulsion system was activated after 50 s in the simulation. The results show that by applying a suited overall control system, the shaft generator and the battery system in combination are able to reduce the transient wave induced loads on the shaft since the amplitude of the speed oscillations are about halved and, hence, smooth the operational conditions for the two stroke engine. This can also be verified by looking at Figure 8.8 which shows the produced diesel engine torque, the torque from the shaft generator (PTI/PTO) and the resulting propeller torque. The diesel engine torque is reduced significantly when the shaft generator is activated. The figure also shows that the produced torque from the shaft generator oscillates around zero, which is as expected due to the control scheme. Also

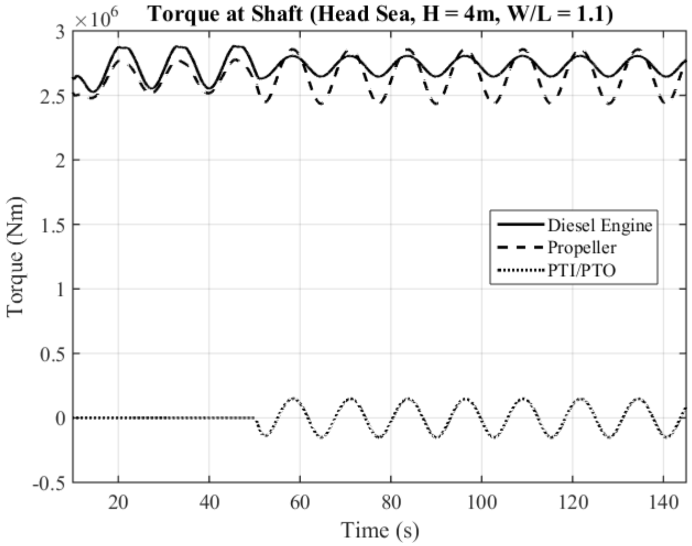


Figure 8.8: Simulation results showing the torque contributions from the main engine and the shaft generator, in comparison to the total torque applied to the propeller. The simulation results are obtained from [43]

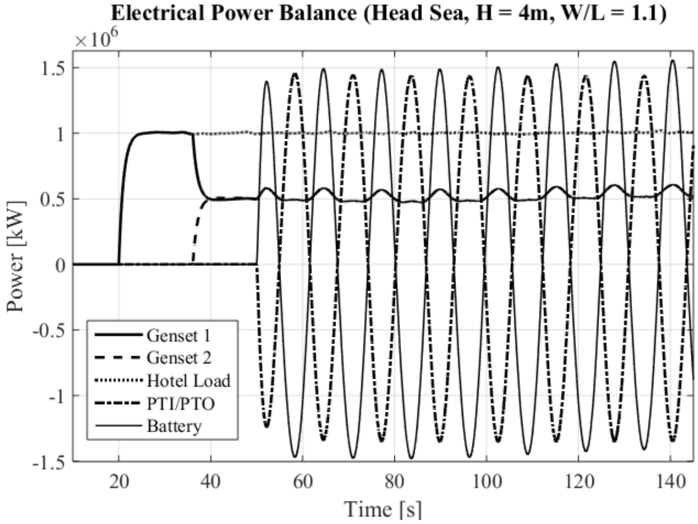


Figure 8.9: Comparison of electrical power balance between the hybrid propulsion system and the conventional propulsion system. The simulation results are obtained from [43]

note that the amplitude of the propeller torque oscillations is increased when the shaft generator is activated. This because the shaft generator is much faster than the main engine and is able to compensate for speed oscillations faster than the main engine.

Figure 8.9 shows the electrical power balance in the co-simulation. As can be seen in the figure, the two gensets in the power plant are not affected too much by the shaft generator and the battery system, but help charging the battery system, due to conversion losses, and can be seen as small bumps in the genset power outputs. Also note that the shaft generator, denoted PTI/PTO (Power Take In/Power Take Off) in the figure, has an opposite power sign in comparison to the battery system, which means that the battery system is charging when the shaft generator takes out power from the shaft (PTO mode).

The results shown in Figure 8.7 and Figure 8.9 are further commented and explained in [43] along with many other simulation results, and will not be given any more attention here. Also, the total system was implemented as one simulation model for comparison reasons, and the conclusion were that about the same simulation results were obtained but the co-simulation was significantly faster to solve the total system, even though a debug-version of Coral was used as the co-simulation master algorithm.

8.3 Optimizing System Integration

When it comes to optimizing a vessel's performance the largest potential lies in system integration and is often related to control system integration on different layers [40]. One typical case is to tune the DP-controller, filters, thrust allocation algorithm and local thruster control systems such that the performance of the total control system result in a fast and stable response of the vessel that minimizes the power consumption and possible distortions. Since such control layers in real life are affected by sampling dynamics and sampling delays, the use of co-simulation works perfectly in simulating the interaction between the different parts of the total control system. One such case study of an offshore marine vessel in DP-operations is presented in the following.

8.3.1 System Overview

The total co-simulation system of the offshore marine vessel in DP-operations is shown with all its submodels and connections in Figure 8.10. As can be seen in the figure, the total co-simulation system consists of 15 FMUs and include a vessel model, propulsor models, propulsor drives, a wave filter, a marine power plant, PID-controllers controlling the propulsor drives, a thrust allocation algorithm, a position and orientation reference system and a DP-controller. In addition, 12 monitoring FMUs are included in the system for displaying simulation results online during the co-simulation. These monitoring FMUs have been implemented

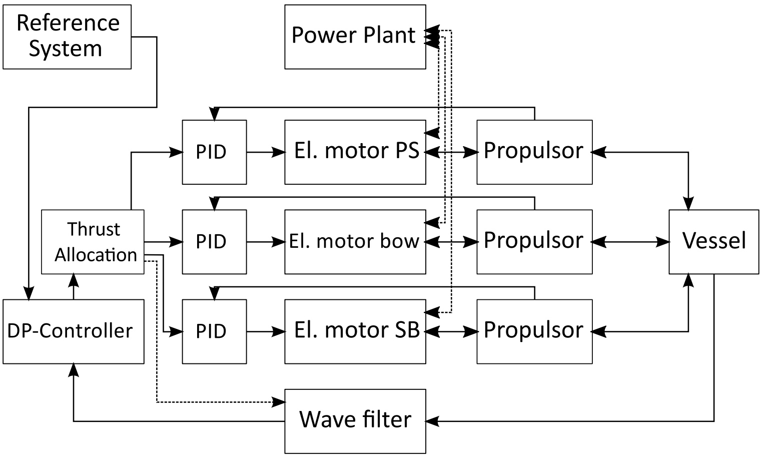


Figure 8.10: Overview of vessel in DP operation including power plant and thruster configuration

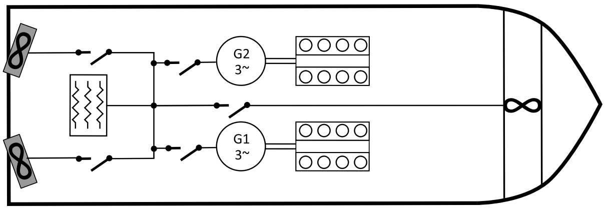


Figure 8.11: System overview of vessel in DP-operation including power plant and thruster configuration

in the C++ programming language with an interface to the Python programming language and the matplotlib library. In the following, a short presentation of the different subsystems in the co-simulation system are given.

Hull and Propulsor Models

The hull model used in this demonstrator is the one derived in [38], see Chapter 5, with the main parameters as given in Table 8.7, also including parameters describing the environment. Note that no crane loads are considered in this case study and that the propulsion system is changed in comparison to the system presented in Chapter 5. Here, two azimuth thrusters placed symmetrically at the stern constitute the main thrusters while the tunnel thruster in the bow is left unchanged, as shown in Figure 8.11. In comparison to the complete vessel model described in Chapter 5, the thrusters, the thrust allocation, the wave filter, the reference system and the DP-controller are removed from the vessel model and added as separate subsystems.

Table 8.7: Main parameters in vessel model

Parameter	**Description**	**Value**
m	Mass of vessel	2365000 kg
L	Length of ship	107 m
B	Width of ship	22 m
D	Draught of ship	5 m
$P_{m,max}$	Maximal power main thrusters	3.5 MW
$P_{b,max}$	Maximal power bow thruster	3.5 MW
v_{cN}	Northward current	-0.1 $^{m}/_{s}$
H_s	Significant wave height	1.0 m
T_p	Wave peak period	8 s
N_w	Number of wave components	50 -
γ	Jonswap-spectrum parameter	3 -
T_d	Lower wave spectra period	0.2 s
T_u	Upper wave spectra period	50 s

The propulsor models include both the propeller inertia and the propulsor drive inertia and the propeller dynamics are as described in section 5.5.4, see Figure 5.10, except that the **MSe** elements have been replaced by propulsor drives which are treated as separate subsystems in the co-simulation.

Power Plant, Propulsor Drives, and Controllers

The power plant model used in this case study is the same as derived in Chapter 6, where the generator models are hybrid causality models, see Chapter 4, except that three more circuit breakers are added in order to interface the propulsor drives as shown in Figure 8.11. Also, the same parameters as presented in Chapter 6 are used.

The propulsor drives are asynchronous electrical motors and are implemented as the shaft generator presented in section 8.2.1, using the same parameters. Each propulsor drive is controlled as in section 8.2.1, see Figure 8.3, where the torque is controlled through the u_{qs} component and the rotor flux is controlled through the u_{ds} component. The torque reference is calculated by an external PID-control law, as illustrated in Figure 8.10, that compares the commanded thrust force from the thrust allocation algorithm and the produced thrust force, which are here assumed measurable, for simplification reasons. This external PID-control law has been implemented in the C++ programming language and compiled as a stand-alone FMU. Note that a reduction gear is also placed between each propulsor drive and propulsor with a gear ratio of 8:1. This, because the rated speed of the propulsor drives is about 1200 RPM while the propulsors run with much lower speeds.

Wave Filter, Reference System, Thrust Allocation and DP-Controller

A wave filter is added to the co-simulation in order for the DP-Controller to only compensate for environmental forces that make the vessel drift, and not the environmental forces that make the vessel oscillate in position and orientation. If these oscillatory forces were to be compensated for a tremendous amount of power would have been required. The wave filter, which is a non-linear passive observer (NLPO), and the DP-control law, which is a simple PID-control law including the heading rotational matrix, are the same as used in Chapter 5, see section 5.5.5.

The reference system is implemented as a separate subsystem in this case study where single predefined way-points are given as user input consisting of a coordinate, an orientation and the time when the desired position and orientation should be activated and when it should be reached, as proposed in [41]. For example, at time t_0 a way-point command is given as (N_c, E_c, ψ_c, t_c) where (N_c, E_c) is the north-east commanded position, ψ_c is the commanded orientation and t_c is the time when the commanded position and orientation should be reached, giving vessel $t_c - t_0$ seconds to respond to the command. A simple interpolation between way-points are conducted in the reference system in order to obtain continuous reference signals. For example, for each commanded coordinate and orientation i,

$$r_{c,i} = [N_{c,i}, E_{c,i}, \psi_{c,i}] \tag{8.10}$$

the reference system calculates the rates of the reference signals as

$$\frac{dr_d}{dt} = \frac{r_{c,i} - r_d}{t_{c,i} - t} \tag{8.11}$$

where r_d is the reference system position and orientation output vector, t is the actual simulation time and $t_{c,i}$ is the time when the desired way-point should be reached. Note that this calculation is not initiated before $t = t_0$. Also, logics are added in order to not divide by zero when $t_{c,i} = t$. The reference system integrates this position and orientation rate vector and the results are filtered with low-pass filters before given as subsystem output to the DP-control system. Note that also the rates are fed to the DP-controller in order to properly control the damping, since the position and orientation rates of the vessel are estimated by the wave filter and fed to the DP-controller as well.

The thrust allocation used in this case study is the same as derived in Chapter 7. Note that in order to increase the response of the azimuth thrusters, thruster biasing for the two main thrusters is considered, such as in Chapter 7. In this case the biasing angle was set to $\pm 20\,^{\circ}$, meaning that if the main thrusters are to produce

Table 8.8: Way-points used in the co-simulation. Note that "-" means that the reference system keeps its old reference value for the respective position coordinate or orientation

t_0 [s]	t_c [s]	N_c [m]	E_c [m]	ψ_c [rad]
0	0	0	0	0
500	700	20	-	-
800	1200	-	-	$\frac{\pi}{2}$
1300	1500	-	20	-
1600	2000	-	-	0
2100	2300	40	-	-
2400	2800	-	-	$-\frac{\pi}{2}$
2900	3300	-	-20	-
3400	3800	-	-	0
3900	4100	60	-	-
4200	4600	-	-	$\frac{\pi}{2}$
4700	4900	-	0	-
5000	5400	-	-	0
5500	5700	80	-	-

thrusts only in the surge direction, one thruster has a biasing angle of $-20\,^\circ$ while the other a biasing angle of $20\,^\circ$.

8.3.2 Simulation Setup

The co-simulation setup is similar to the one presented in section 7.5, except that now the power plant dynamics and the propulsion drive dynamics are included in the study. In the co-simulation the auxiliary power grid load is set to $100\,kW$ in addition to some power grid load variations modeled as noise, representing a low hotel load or auxiliary load for the vessel. The vessel is to move in a square-like pattern in the co-simulation, the same as in section 7.5, and the respective way-points are given in Table 8.8. Two different tuning cases of the thrust allocation algorithm are considered here and are the same as Case 1 and Case 3 as presented in Table 7.6. The global communication time-step in the co-simulation is set to 10 ms. However, note that the DP-controller only communicated with the connected systems every second. The connections between the subsystems in the co-simulation are shown in Table 8.9 and the integration methods for the different subsystems as well as the local solver time-step sizes are given in Table 8.10.

The total length of the co-simulation is set to $6000\,s$. Note that the DP-controller has been retuned in comparison to the values used in Chapter 4 since it communicates with the rest of the subsystems with a much lower frequency.

Table 8.9: Connections between subsystems. Note that the connecting variables between subsystems are given in SI-units except for $\mathcal{M}$,$\mathcal{R}$ and τ which are abbreviations for measurement, reference and thrust forces and torques, respectively

CONNECTIONS	**Input**	**Vessel**	**Propulsor**	**El. motor**	**Power plant**	**PID**	**Thrust alloc.**	**DP**	**Wave filter**
Output	⤴	↑	↑	↑	↑	↑	↑	↑	↑
Vessel	→		m/s						$\mathcal{M}$
Propulsor	→	N, rad		rad/s		$\mathcal{M}$			
El. motor	→		Nm		A				
Power plant	→			V					
PID	→			$\mathcal{R}$					
Thrust alloc.	→					$\mathcal{R}$			τ
DP	→						$\mathcal{R}$		
Wave filter	→							$\mathcal{M}$	
Ref. system	→							$\mathcal{R}$	

Table 8.10: Subsystems and integration methods

Subsystem	**Integration method**	**Time step**
Hull	Runge-Kutta, 2st order	1 ms
Propulsors	Euler, 1st order	1 ms
Propulsor drives	Runge-Kutta, 4th order	0.05 ms
Power Plant	Euler, 1st order	0.1 ms
Non-linear passive observer	Runge-Kutta, 2st order	5 ms
PID-controllers	Euler, 1st order	10 ms
Reference system	Euler, 1st order	10 ms
Thrust allocation[1]	Euler, 1st order	10 ms
DP-controller[2]	Euler, 1st order	10 ms

8.3.3 Simulation Results

In this case study the main focus is given to simulation results regarding the interactions between the DP-controller, the thrust allocation algorithm, the power plant and the propulsion drives. However, the simulation results regarding the position

[1] The optimization in the thrust allocation is only initiated every second

[2] The DP-controller only updates its input and output values every second

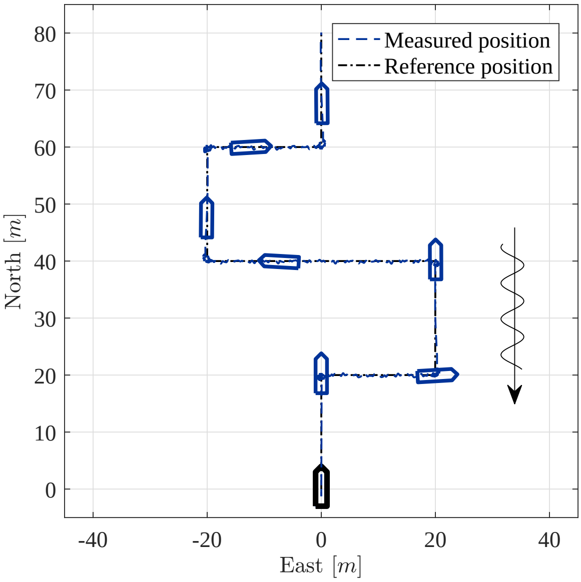


Figure 8.12: North-east position and heading of vessel in square wave trajectory manoeuvre. Black vessel outline in the plot denotes initial position and orientation

and the orientation of the vessel are given in Figure 8.12 and Figure 8.13. The simulation results showing the vessel in a north-east plot compared to the desired position overlap in the two tuning cases and are given in Figure 8.12. As can be seen in the figure, the vessel follows its reference quite well, even though there are more noise on the position of the vessel when the vessel faces the waves with the heel. Note that in each corner in the position trajectory the vessel changes heading while trying to keep a fixed north-east position. This is verified in Figure 8.13 which compares the vessel's position and orientation with the commands from the reference system. As can be seen in the figure the vessel oscillates a bit around its reference position and orientation. This is because the wave filter filters out these fast oscillations before feeding the DP-controller with the measurements, which is desired because the vessel is not able to compensate for these fast oscillations and if they were fed to the DP-controller it would only result in a much higher power consumption and possibly an unstable DP-controller. Note that all these results are obtained for tuning case 1 but overlap with the results from case 2.

The simulation results from the propulsion system as well as the power plant are shown in Figure 8.14 for the first tuning case. The first plot in the figure shows the azimuth angles for the two main thrusters at the stern. As can be seen, the thruster angles stays between $\pm 180\,^{\circ}$ and one can clearly see the thruster biasing

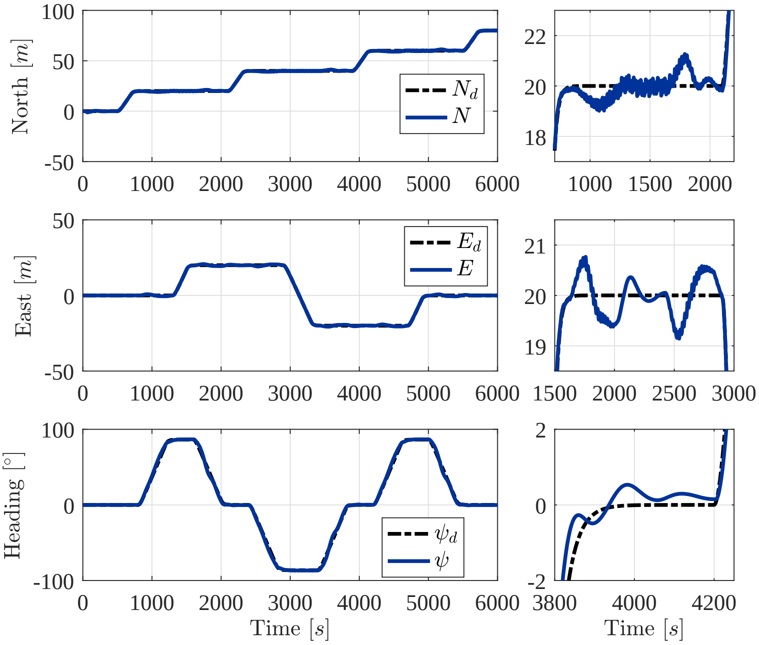


Figure 8.13: Vessel position and orientation compared to the position and orientation commands

angle for the two thrusters. The second plot shows the thrust produced by the two azimuth thrusters. The three regions with high values and a lot of noise are because the vessel moves in the east- or west direction, facing the waves with the heel. The third plot shows the thrust produced by the bow thruster. Also here, there are some oscillations present due to the wave effects. The last plot in the figure shows the produced power by generator 1 (G1) and generator 2 (G2), which overlap, and the total vessel power consumption. Since the produced thrusts from the three thrusters oscillate when the waves encounter the heel of the vessel, it is not surprising that the power consumption oscillates as well. However, by tuning the different control systems properly altogether it is possible to obtain a smoother power consumption as well as smoother operation of the thruster systems. This has been done and the corresponding results are shown in Figure 8.15.

As can be seen in the figure, both the azimuth angles and the produced thrusts oscillate less in this case in comparison to the previous one, when neglecting the initial oscillations for the azimuth angles and the corresponding thrusts. Also, the oscillations in the power consumption are reduced and result in a slightly lower power

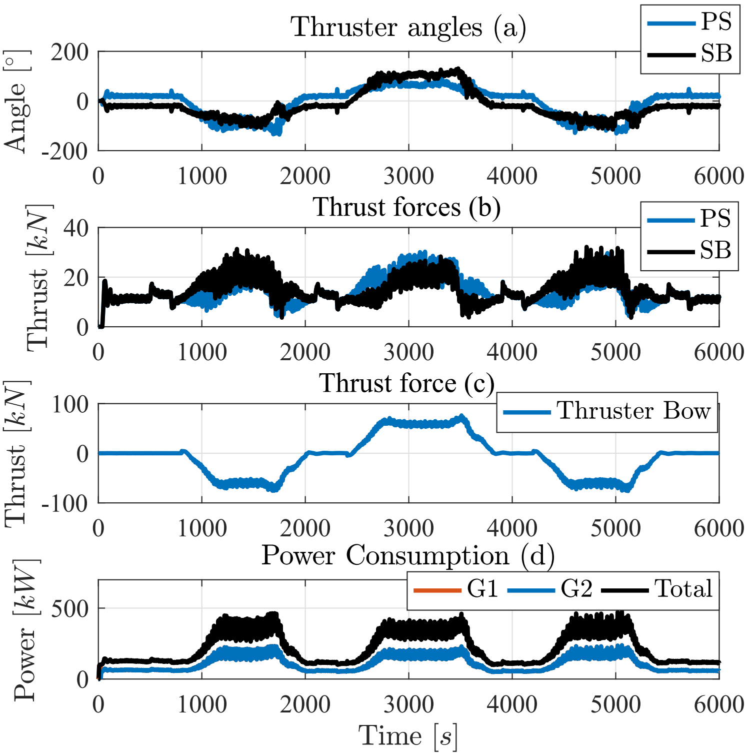


Figure 8.14: Simulation results showing the thruster azimuth angles for the two main thrusters placed at the stern, the corresponding thrust, the thrust of the bow thruster and the power produced by the two generators as well as the total power consumption for the first tuning case.

consumption as well as reducing wear of the propulsion system and the power plant. The total energy consumption from tuning case 1 was about 367.61 kWh in comparison to the results from section 7.5 where the energy consumption for the same case was 200.9 kWh. These results are comparable when subtracting the hotel load from the total energy consumption, which means that the propulsion system consumed about 200.94 kWh. However, since the DP-controller has been retuned here in comparison to the simulation results shown in section 7.5 we can not compare the results directly. Since the propulsor drives have internal losses we can conclude from the results that the DP-controller is not tuned as hard in this case study in comparison to the case study in section 7.5, which is true. The energy consumption data is summarized in Table 8.11 for both tuning cases.

The Brake Specific Fuel Consumption (BSFC), the Instantaneous Fuel Consump-

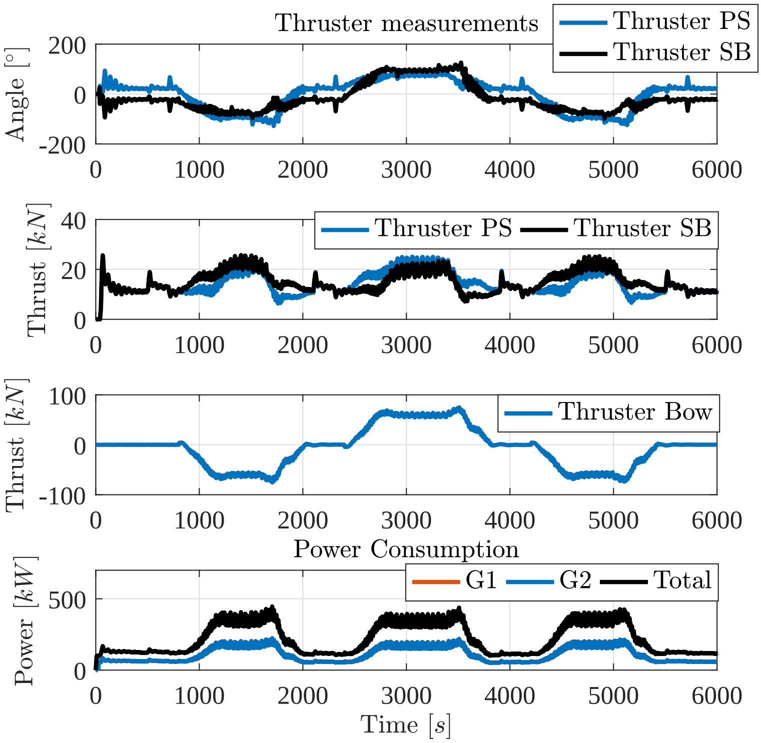


Figure 8.15: Simulation results showing the thruster azimuth angles for the two main thrusters placed at the stern, the corresponding thrust, the thrust of the bow thruster and the power produced by the two generators as well as the total power consumption for the second tuning case.

Table 8.11: Energy consumption

Case	Hotel load (mean)	Propulsion system	Total
1	166.67 kWh	200.94 kWh	367.61 kWh
2	166.67 kWh	200.80 kWh	367.47 kWh

tion (IFC) and the fuel efficiency for the auxiliary engines driving the generators in the power plant for the two cases are compared in Figure 8.16. The first plot in the figure shows the specific fuel consumption for genset 1 in both tuning cases in additional to the best BSFC the auxiliary diesel engine can obtain, which in this case is at a loading of about 90 %, equivalent to about 1810 kW, having a BSFC of about 193 g/kWh. The results show that the BSFC for the auxiliary engines overlap but tuning case 2 has a lower amplitude for the oscillations. Also, the results sug-

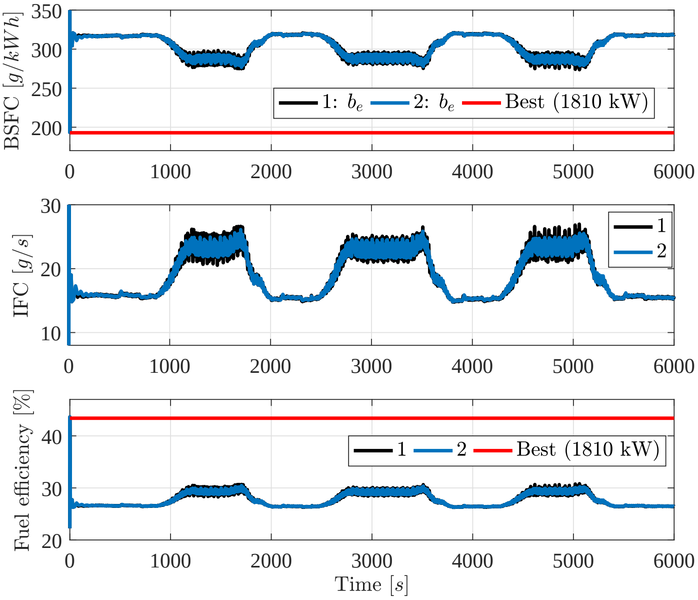


Figure 8.16: Comparison of the Brake Specific Fuel Consumption (BSFC), the Instantaneous Fuel Consumption (IFC) and the fuel efficiency for the two tuning cases, in comparison to the best possible results

gest that the gensets are running with relatively low load since the BSFC is high, which can be verified by the fact that each auxiliary engine has a maximal loading of 16.9 % or 17.6 %, depending on the tuning case, and a mean loading of about 11.3 %.

The second plot in the figure shows the instantaneous fuel consumption for genset 1 in both tuning cases. Also here the results overlaps and tuning case 2 has slightly lower amplitude of the oscillations. Nevertheless, the fuel consumption were about 227.87 kg in tuning case 1 and 227.91 kg in tuning case 2, meaning that the two cases have about the same fuel efficiency. This is verified in the last plot which shows a comparison of the fuel efficiencies when assuming that the lower heating value of diesel is given as 42.7 MJ/kg. The results show that in both cases the fuel efficiency is about 27.7 % in mean value, in comparison to the best possible fuel efficiency of about 43.4 % that can be obtained for this engine. Since the total power consumption in both co-simulations are below the best loading of the auxiliary engine, the fuel efficiency would have been improved if only one of the gensets in the power plant was active, which would have increased the fuel efficiency with about 5.1 %, saving about 35 kg fuel.

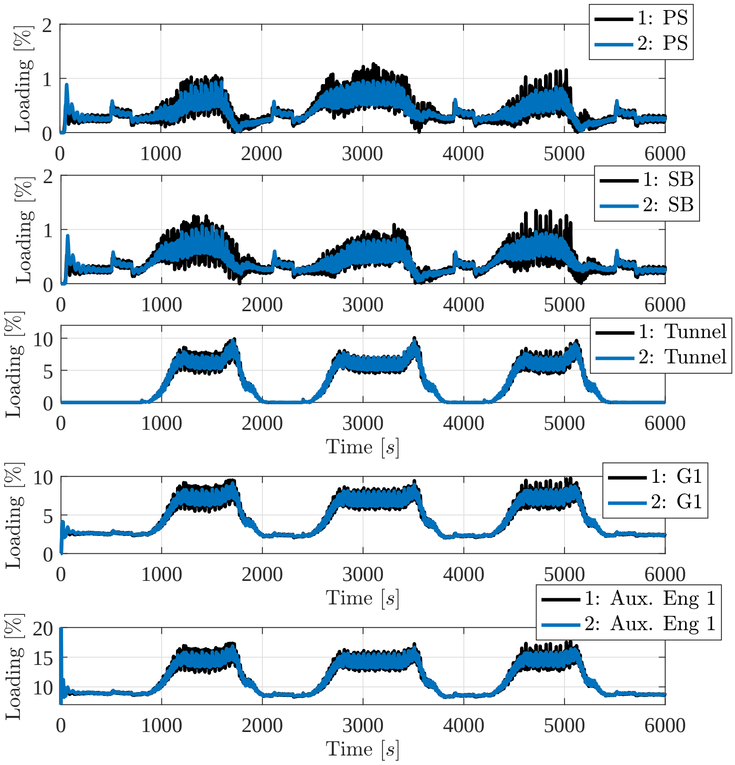


Figure 8.17: Comparison of loading of propulsor drives, generator 1 and auxiliary engine 1 for both tuning cases

To complete this case study, the loading given in percentage for the propulsor drives, generator 1 and auxiliary engine 1 are compared in Figure 8.17 for the two tuning cases. As can be seen in the figure the lading of the propulsor drives on the starboard side (SB) and ones on the port-side (PS) overlap in the two tuning cases. The maximal loading of the propulsor on the port-side in case 1 is about 1.4 %, while the same propulsion drive in case 2 has a maximal loading of about 0.95 %. This is quite low and would result in a poor overall efficiency. Hence, smaller propulsor drives should be considered in a design phase for DP-operations in order to increase the loading and, hence, increase the propulsor drive efficiency. The third plot in the figure shows a comparison of the loading of the propulsion drive for the tunnel thruster for both cases. Also here the results seem to overlap and the maximal loading for case 1 is about 10 % while being 9.7 % for case 2, and happens when the vessel faces the wave front with the heel. The fourth plot shows a comparison of the loading for generator 1 in the two tuning cases. As can be seen in the figure the loadings overlap in the plot and the oscillations are smaller in case 2 in comparison to case 1. Also here the loadings are quite low, having maximal values of about 9.8 % and 9.2 % for case 1 and 2, respectively. Hence, the efficiency would have been improved if only one of the generators were active. The same can be concluded regarding the loading of the auxiliary engine shown in the last plot having maximal loadings of 17.6 % and 16.9 % for case 1 and case 2, respectively. However, regulations often permits the use of only one active genset in DP-operations for safety reasons regarding the risk of black-out. Note that the polynomial used for calculating the BSFC given in (6.49) is not valid for engine loadings below 10 % and as can be seen in the figure, the auxiliary engines are slightly below this limit when the vessel faces the waves with the bow. Hence, the BSFC results in these situations, as well as the corresponding fuel efficiencies, are not to be completely trusted.

The results show that both the power plant and the propulsion drives have low loads which will result in low efficiencies. Nevertheless, both the generator size and the propulsor drive size for the main thrusters are needed when the vessel is in transit. For example, the vessel model derived in Chapter 5 has a maximal speed of about 15.5 kn which is obtained when both main propulsors in Chapter 5 run at full speed, consuming about 3.5 MW each. Such low loadings obtained in the co-simulations are also normal in real DP-operations in calm weather conditions for such offshore support vessels. Hence, it is a design challenge to design the propulsion system and the marine power plant for transit operations as well as DP-operations in calm weather conditions. However, as have been shown in this case study, co-simulations can be used for evaluating multiple vessel configurations and for optimizing the vessel for its operations.

In such a co-simulation one crucial requirement for the control system to perform properly is the choice of sampling frequency of the different components in the overall vessel control system, and is strongly related to both dynamical stability of the total system and combined stability of the entire co-simulations, see Chapter 3. In general, each control layer should be tuned such that the outer control layers are slower than the inner control layers. They may also have a lower sampling frequency. Here, the outer control layer consists of the DP-controller and the thrust allocation algorithm, which both have a sampling frequency of 1 Hz, while the inner control layer consists of the wave filter and the local thruster control systems, having sampling frequencies of 100 Hz. Note that care must be taken when tuning the DP-controller since it contains integration effects and since it has such a low sampling frequency. Hence, such a total system analysis using co-simulations is

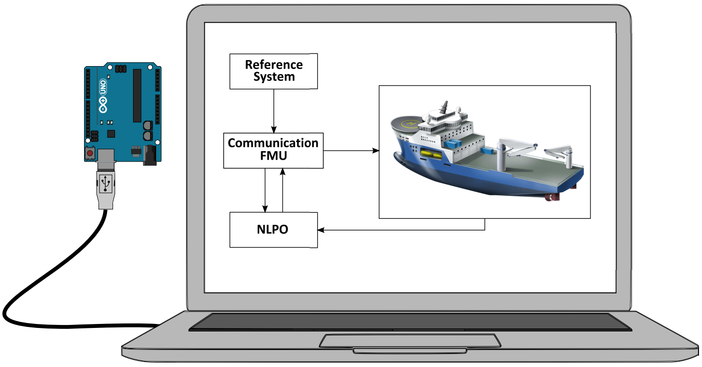


Figure 8.18: Co-simulation setup using hardware in the simulation loop

also well suited for tuning and verifying control systems.

In case studies related to control systems it is also of interest to include hardware in the simulation loop. This is discussed in the following where a simplified version of the case study presented here is used and where the DP-control system has been implemented on a micro-controller.

8.4 Hardware-In-the-Loop (HIL)

A small case study including Hardware-In-the-Loop (HIL) in co-simulations were initiated in this thesis work [41]. When including hardware in a simulation loop, proper communication between the hardware and the simulation is important. Because the FMI-standard has predefined functions that are called by the simulation master algorithm, such as the function *fmiDoStep*, it is possible to make a FMU with suited functionality such as reading and sending data in a consistent manner through a serial port on the computer running the co-simulation, as will be elaborated in this case study. Here, the focus is given to the interaction between the DP-controller and the wave-filter, hence, simplified thruster models and a static thrust allocation were implemented directly into the vessel model, the same as derived in [38], in comparison to the system studied in section 8.3. An overview of the subsystems used in the co-simulation is given in the following.

8.4.1 System Overview

The total co-simulation system includes a vessel model with embedded thrust allocation and propulsion system, a wave filter, a reference system and a communication system for communicating with the hardware, as shown in Figure 8.18. In the figure, the vessel's position and orientation measurements are fed to the wave filter, which filters the measurements and estimates the measurement rates. These filtered and estimated states are then fed to the communication FMU. Also, position references, orientation references and corresponding rates are fed to the communication FMU by the reference system. Then, the communication FMU sends both measurements and references to the micro-controller, with the DP-control law embedded, which calculates the new global thrust commands. These commands are then sent back to the communication FMU and forwarded to the vessel model. The different subsystems in the co-simulation are presented in more detail in the following.

Vessel Model, Wave Filter and Reference System

The vessel model used in this co-simulation is the same as the one derived in Chapter 5, see Figure 5.6, except that the DP-control system, the reference system and the wave-filter, all in which are incorporated in the *"DP control system"*-block in the figure, are separate subsystems in the co-simulation. Note that the crane system are here assumed to have no payload during the entire co-simulation. The main vessel parameters are as given in Table 8.7 except that now the significant wave height is set to 1.5 m and the northward current is set to -0.2 m/s. Note that both the propulsion system and the thrust allocation are as presented in section 5.5.4.

The wave filter is the same as used in section 8.3, a non-linear passive observer as described in section 5.5.5. Also the reference system is the same as the one presented in section 8.3. Note that only North-East reference positions are considered since a static thrust allocation and fixed positioned main thrusters are considered. Hence, the vessel is always oriented such that it faces the wave front which comes from the north.

DP-Controller and Communication FMU

The DP-controller is the same as the one used in section 8.3 and the controller gains are given in Table 8.12. The DP-control law is implemented on an Arduino micro-controller [211] and connected through a serial port to the computer running the co-simulation, as illustrated in Figure 8.18.

The communication FMU, that connects the Arduino UNO controller to the co-simulation, has the property of routing signals between the hardware and the simulation at given time instances. Since superior controllers, such as DP-controllers, operate with lower frequencies than the ones required to obtain stable simulation results in the co-simulation, the communication FMU should have the property of only synchronizing the hardware DP-controller with the co-simulation at local time steps larger than or equal to the global communication time step, while keeping all signals constant between. Code 8.1 shows a pseudo-code explaining how

Table 8.12: Controller gains used in the "HIL" co-simulation

Proportional gain north	22000 N/m
Proportional gain east	20000 N/m
Proportional gain heading	4000000 N/rad
Derivative gain north	200000 Ns/m
Derivative gain east	150000 Ns/m
Derivative gain heading	40000000 Ns/rad
Integral gain north	500 N/sm
Integral gain east	500 N/sm
Integral gain heading	5000 $N/srad$

the functionalities in the communication FMU can be implemented in the FMI-standard.

Code 8.1**:** Pseudo-code of serial communication FMU (C++ programming language syntax)[41]

```
fmiStatus fmiInitializeSlave(...,$t_{start}$){
        $t_l = t_{start}$;
        SetInit(...);
        OpenSerial(...);
        return fmiOK or fmiError;// Status check needed
}
fmiStatus fmiSetReal(...,Input){
        $S$ = Input;
        return fmiOK;
}
fmiStatus fmiDoStep(...,$t_g$){
        if($t_l$ + $\Delta t_c < t_g$){
                SerialWrite($S$);
                SerialRead($\tau_c$);
                $t_l = t_l + \Delta t_c$;
        }
        return fmiOK or fmiError;// Status check needed
}
fmiStatus fmiGetReal(...,Output){
        Output = $\tau_c$;
        return fmiOK;
}
fmiStatus fmiTerminateSlave(...){
        CloseSerial(...);
        return fmiOK or fmiError;// Status check needed
}
```

In the pseudo-code, t_l is the local propagating time in the communication FMU,

Table 8.13: Function description for functions regarding the communication FMU

Function	Description
`SetInit(...)`	Initializes the communication FMU.
`OpenSerial(...)`	Opens and establishes serial connection to the hardware controller.
`SerialWrite(`$\boldsymbol{S}$`)`	Sends the information stored in $\boldsymbol{S}$ to the hardware controller.
`SerialRead(`$\boldsymbol{\tau}_c$`)`	Reads the information sent by the hardware controller and stores it into $\boldsymbol{\tau}_c$.
`CloseSerial(...)`	Terminates the serial connection to the hardware.

Table 8.14: Way-points used in the co-simulation. Note that "-" means that the reference system keeps its old reference value for the respective position coordinate or orientation

t_0 [s]	t_c [s]	N_c [m]	E_c [m]	ψ_c [rad]
0	0	0	0	0
100	300	50	-	-
400	800	-	50	-
900	1100	0	-	-
1200	1400	-	25	-
1500	1600	25	-	-

t_g is the global propagating co-simulation time, t_{start} is the starting time of the simulation (typically $t_{start} = 0$), Δt_c is the local time step in the communication FMU, S is the data vector that is sent from the communication FMU to the hardware, τ_c is the commanded global thrust vector sent from the hardware in feedback, and `Input` and `Output` are the input and output vectorial communication ports connecting the communication FMU to the rest of the co-simulation, respectively. Note that functions and types defined in the FMI-standard are given with the color light blue, while functions given in green has been implemented in this work and are explained more in detail in Table 8.13. It should also be mentioned that functionalities regarding security checks and allocation- and deallocation of memory are also implemented, but omitted in this presentation.

8.4.2 Simulation Setup

In the simulation the vessel is to face the encountering waves while moving in a square-like trajectory, meaning that the heading reference is always zero while the north-east references change. The corresponding way-points are shown in Table 8.14 and the connections between the subsystems in the co-simulation are shown in Table 8.15. In the co-simulation the global communication time step is set to

Table 8.15: Connections between subsystems. Note that the connecting variables $\mathcal{M}$, $\mathcal{R}$, $\mathcal{S}$ and τ are abbreviations for measurement, reference, data string and thrust forces and torques, respectively

CONNECTIONS	**Input**	**Vessel**	**Hardware**	**Comm. FMU**	**Wave filter**
Output	⤴	↑	↑	↑	↑
Vessel	→				$\mathcal{M}$
Hardware	→			$\mathcal{S}$	
Comm. FMU	→	τ	$\mathcal{S}$		τ
Wave filter	→			$\mathcal{M}$	
Ref. system	→			$\mathcal{R}$	

Table 8.16: Subsystems and integration methods

Subsystem	**Integration method**	**Time step**
Vessel	Euler, 1st order	10 ms
Non-linear passive observer	Euler, 1st order	10 ms
Reference system	Euler, 1st order	10 ms
Communication FMU	Euler, 1st order	1000 ms

50 ms, the local time step for each subsystem in the co-simulation is listed in Table 8.16 along with the corresponding integration method and the total co-simulation time is set to 2000 s. A selection of the co-simulation results are presented in the following.

8.4.3 Simulation Results

The simulation results regarding the position and orientation of the vessel are shown in a north-east plot in Figure 8.19. As can be seen in the figure, the vessel seems to keep its position and orientation also in this case when the DP-controller is placed on a micro-controller. This is verified in Figure 8.20 which compares the measured vessel position and orientation (N,E and ψ) with the filtered ($\hat{N}$,$\hat{E}$ and $\hat{\psi}$) and commanded positions and orientations (N_d,E_d and ψ_d). Note that it takes some time to update the biases in the wave filter and is why the vessel orientation does not converge to the commanded orientation before about 1500 s has passed. However, the error is quite low, about 0.17 ° at most. The corresponding rates for the north-east position, as well as the heading, estimated by the NLPO ($\dot{\hat{N}}$,$\dot{\hat{E}}$ and

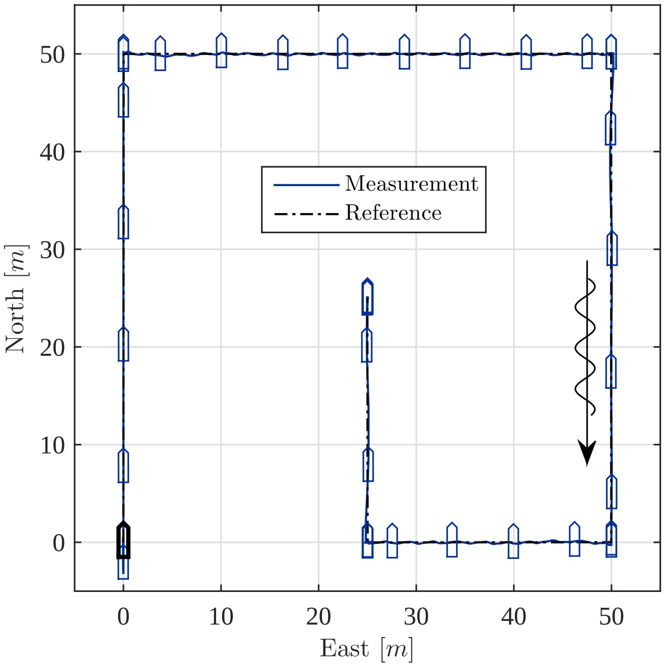


Figure 8.19: North-east position and heading of vessel in DP-operation. Black vessel outline in the plot denotes initial position and orientation

$\dot{\hat{\psi}}$) are compared to the actual rates ($\dot{N}$,$\dot{E}$ and $\dot{\psi}$) and the reference rates ($\dot{N}_d$,$\dot{E}_d$ and $\dot{\psi}_d$) in Figure 8.21. The first plot in the figure compares the estimated north rate with the actual north rate and the desired north rate, the second plot compares the estimated east rate with the actual east rate and the desired east rate while the last plot shows the estimated heading rate compared to the actual heading rate and the desired heading rate. As can be seen in the figure the wave-filter is able to filter out most of the wave induced oscillations as well as generating good position and orientation rates. It can also be seen from the figure that the rates have biases in the beginning of the simulation. This has also to do with the fact that the wave-filter needs some time to update the biases representing the slowly varying drift forces caused by the second order wave effects and the current. Nevertheless, the results show clearly that the wave-filter works properly. Figure 8.22 shows the commands from the DP-controller in north, east and yaw for the vessel.

The first plot in the figure shows the commanded thrust force in surge, the second plot shows the commanded thrust force in sway and the last plot shows the commanded thrust torque in yaw. As can be seen in the figure, the DP-controller seems to be stable and control the vessel to its desired position even though being imple-

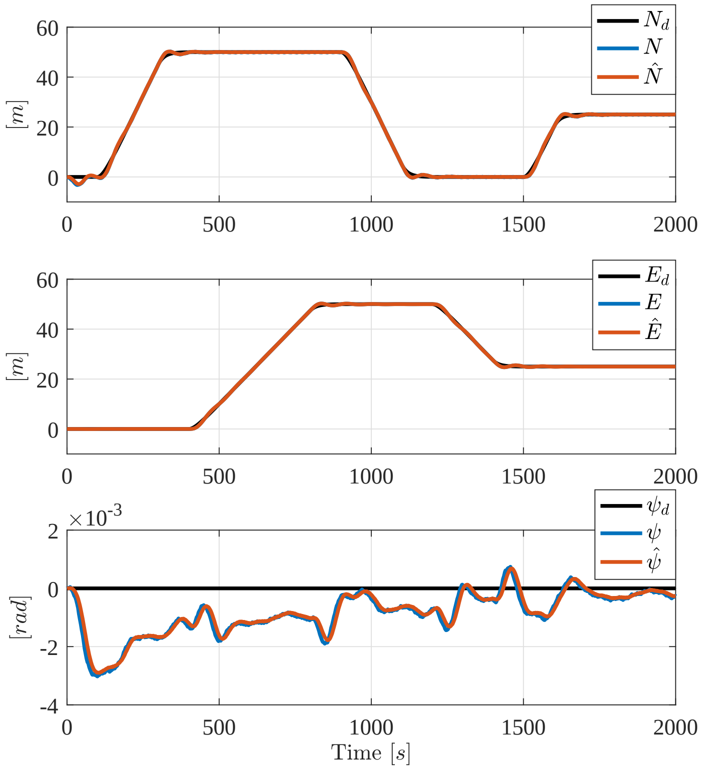


Figure 8.20: The measured vessel positions and orientations (N,E and ψ) compared to filtered ($\hat{N}$,$\hat{E}$ and $\hat{\psi}$) and commanded vessel positions and orientations (N_d,E_d and ψ_d)

mented on a micro-controller and only able to communicate with the rest of the simulation every second. This means that the DP-controller implemented on the micro-controller has the same characteristics and the same sampling properties as the DP-controller implemented as an FMU in section 8.3. Hence, the only difference of any significance is the communication protocol used to communicate with the micro-controller as well as real-time limitations related to connecting hardware to the co-simulation environment.

This case study shows that hardware can be included in co-simulations in a generic way by making a generic FMU that has the only task of being the communication link between the hardware and the co-simulation environment. Hence, it would be straight forward to replace the DP-controller in section 8.3 with the communication-FMU and the micro-controller. One of the good features with co-

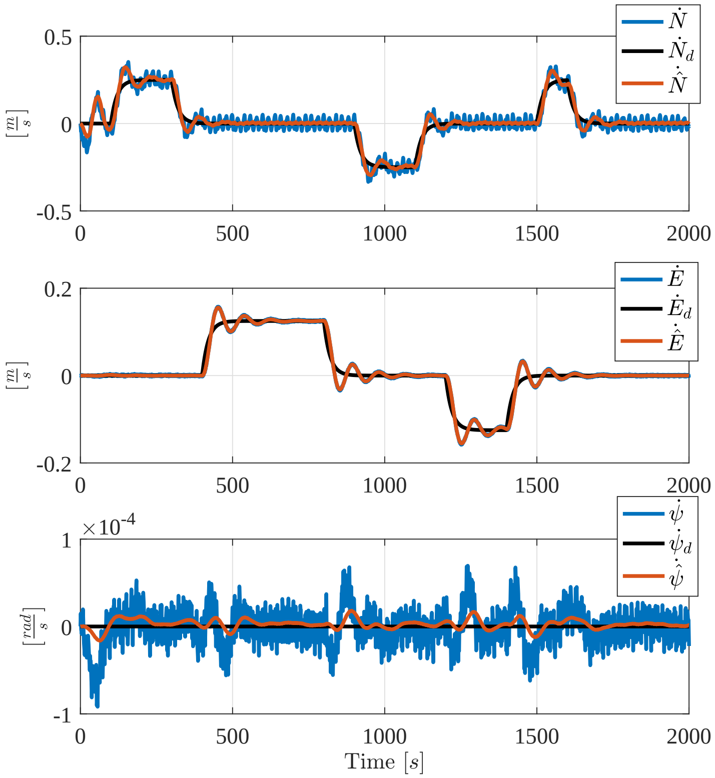


Figure 8.21: Simulation results coparing the estimated north-, east- and heading rates ($\dot{\hat{N}}$,$\dot{\hat{E}}$ and $\dot{\hat{\psi}}$) with the actual rates ($\dot{N}$,$\dot{E}$ and $\dot{\psi}$) and the commanded rates ($\dot{N}_d$,$\dot{E}_d$ and $\dot{\psi}_d$)

simulations is that it is simple to replace subsystems in a co-simulation with other subsystems in a modular fashion. This is especially beneficial when designing new vessels where one would like to test different vessel configurations and equipment on beforehand in a virtual setting before actually building the vessel. Such a case is presented in the following.

8.5 Testing different Vessel Configurations

NTNU's research vessel *R/V Gunnerus*, see Figure 8.23, is a multi-purpose vessel used in research project, spanning from developing DP-controllers, autopilots, autonomous vessel operations, sub-sea operations using ROV's, surveillance using UAV's, testing fishing equipment and mapping the seabed. The research vessel is equipped with two main propulsors including rudders placed symmetrically at the

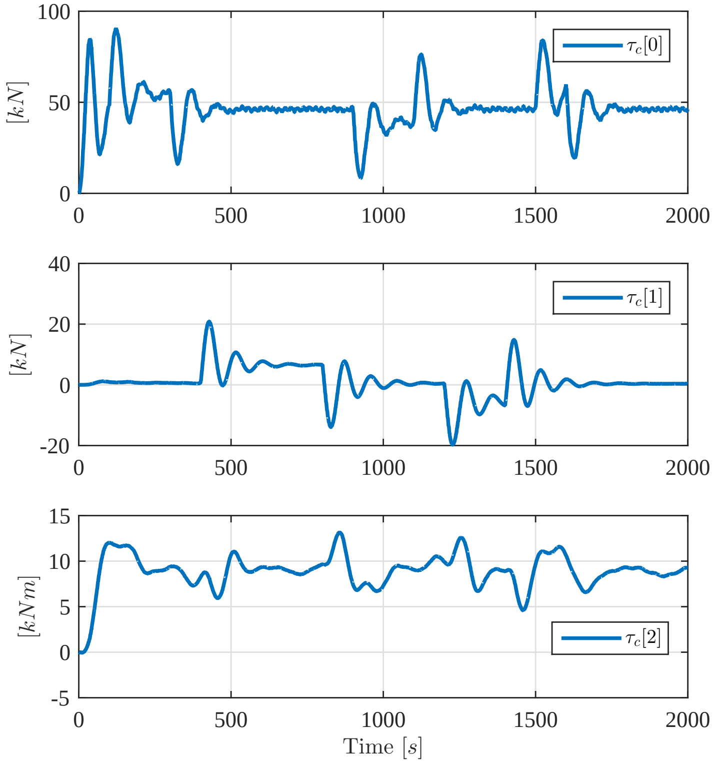


Figure 8.22: DP-controller commands fed to the co-simulation from the micro-controller through the communication FMU

stern and one tunnel thruster in the bow.

In this case study two different main propulsor units for Gunnerus will be tested and compared in a zig-zag test in calm sea –the old configuration including fixed propulsors and rudders and the new configuration including azimuth thrusters. This case study is a joint project between different partners in the ViProMa project and illustrate the use of black-box models in conceptual design studies where different subsystems are made by different vendors.

The main focus in this study is given to the propulsor units and the hydrodynamical performance of the main thrusters. Hence, both the power plant and the propulsor drives are assumed equal in both cases for comparison reasons. An overview of the co-simulation system is given in the following.

Figure 8.23: NTNU's research vessel R/V Gunnerus

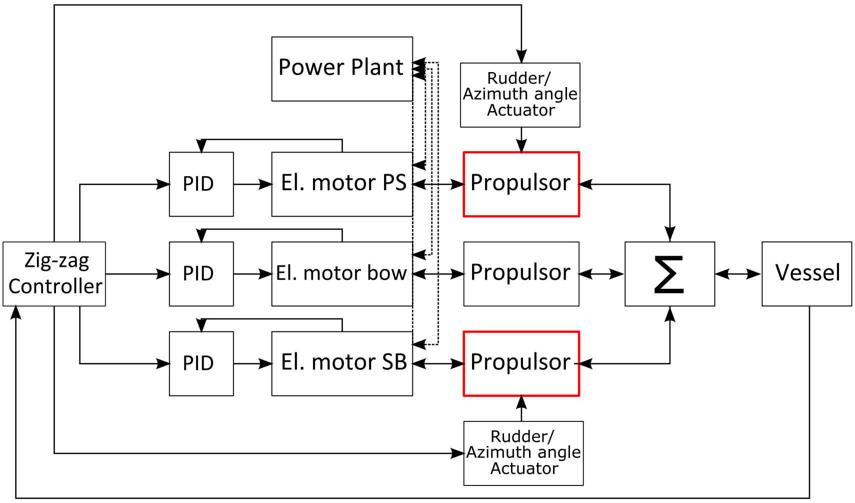


Figure 8.24: Co-simulation setup for the research vessel Gunnerus

8.5.1 System Overview

The total co-simulation setup is as shown in Figure 8.24. In the figure the main propulsors placed at the stern are outlined in red color in order to illustrate that these are the only models that need to be replaced when changing the main propulsors. Some of the models used in this case study are developed by different partners in the ViProMa project, such as Rolls-Royce Marine and SINTEF Ocean, and are delivered as black-box models.

The total vessel model is developed based on the specifications of the vessel and in-house mathematical models obtained from different modeling and simulation

Table 8.17: R/V Gunnerus main parameters

Parameter	**Description**	**Value**
L_{OA}	Overall Length of ship	31.25 m
L_{PP}	Length between perpendiculars	22 m
B	Moulded breadth	9.6 m
dwt	Dead weight	75 t
D	Draft	2.7 m
P	Generator power	3 x 450 kW

Table 8.18: Main parameters for the propulsion units. The case *Old* represents the old main propulsion configuration with fixed propellers and rudders while the case *New* represents the new propulsion configuration with azimuth thrusters

Parameter	**Description**	**Old**	**New**
$\dot{\alpha}$	Rudder rate limit	5.85 °/s	5.85 °/s
D	Propeller diameter	2.0 m	1.9 m
P	Propeller power	2 x 500.0 kW	2 x 500.0 kW
Z	Number of blades	5	4

software. The main parameters describing the vessel are given in Table 8.17. A more detailed presentation of each subsystem in the co-simulation is given in the following.

Hull Model and Propulsor Models

The hull model is a black-box VeSim[3] model delivered by SINTEF Ocean in the ViProMa project that was developed in the "Sea Trials and Model Tests to Validate Shiphandling Simulation Models" (SimVal) project, funded by The Research Council of Norway [212, 21, 9]. This hull model, in contrast to the one derived in Chapter 5 includes more advanced hydrodynamics and are more suited when testing different propulsion units since the hull resistances are crucial for the operation of the propulsion system. The main parameters for the hull model are given in Table 8.17.

The azimuth models was developed by Rolls-Royce Marine in the ViProMa project, while the propeller and rudder models are generic VeSim models developed by SINTEF Ocean, parametrized to fit Gunnerus. Note that all these propulsor models are black-box models. The main parameters for both propulsion units are given in Table 8.18. The contributions from each propulsor unit are here sum-

[3] VeSim is an in-house time-domain simulation tool for the simulation of ships in a seaway with variable heading and speed [9] developed by SINTEF Ocean.

marized in the $\sum$-block in Figure 8.24. Note that this $\sum$-FMU is added in the co-simulation in order to properly calculate the thrust losses which are included in this study, such as when two thrusters are facing each other the produced thrusts are reduced because they affect each others water flows through the propellers. This effect is here referred to as thruster-thruster interactions. Other thrust losses that are included is propeller ventilation and propulsor-hull interactions. Note that these propulsor units are quasi-static, meaning that no numerical solver is needed. Hence, the local time steps for the propulsor unit models, as well as for the $\sum$-FMU, are set equal to the global communication time step.

Zig-Zag Controller, Rudder- and Azimuth Angle Actuator

The zig-zag controller is a black-box model derived in Matlab by SINTEF Ocean and has the main purpose of setting the vessel speed and heading through feeding the propulsion drive controllers and the rudder/azimuth angle actuators with references. Note that the propulsor drives are given different references in the two cases in order to obtain as equal vessel speeds as possible. Also note that the Zig-Zag controller does not contain any equations that need a numerical solver, hence no solver is implemented, and the time step is set equal to the global communication time step.

The rudder and azimuth angle actuators are assumed ideal and implemented as simple rate limiters, meaning that if a step-reference of 10 ° is given to the actuators they ramp the rudder/azimuth angles to the desired value with a maximal rate of $\dot{\alpha}$, here set to ±5.85 °/s. Since the rudder- and azimuth angle actuator subsystems are simple rate-limiters, no numerical solver is needed. Hence, the local time steps are set equal to the global communication time step also for subsystems models.

Power Plant Model, Propulsor Drive Models and PID-Controllers

Since the main focus in this case study is not given to the power plant and the electrical systems the same power plant model as derived in Chapter 6 and the propulsion drives presented in section 8.2.1 are used. Note that both the power plant model and the propulsor drives have much higher capacities than the ones installed in Gunnerus. However, the capacity of Gunnerus will not be exceeded in the simulations presented here. Hence, the power plant model and the propulsor drives are only added in order to complete the total system and for adding realistic interactions. Another argument for using the already derived power plant and propulsion drives is that only a comparison of the performance of the propulsion units are in scope here, and since both propulsion units use the same propulsion drives and the same power plant, the obtained co-simulation results are directly comparable. If the scope in this case study is increased to also include the per-

Table 8.19: Subsystems and integration methods

Subsystem	Integration method	Time step
Hull	Euler, 1st order	10 ms
Propulsor units	quasi-static, no solver	10 ms
$\sum$	static, no solver	10 ms
Rudder/azimuth angle actuators	rate limiters, no solver	10 ms
Zig-Zag controller	only logics, no solver	10 ms
Power plant	Euler, 1st order	0.1 ms
Propulsor drives	Runge-Kutta, 4th order	0.05 ms
PID-controllers	Euler, 1st order	10 ms

formance of the power plant, another more suited power plant model, as well as propulsion drive models, should be considered.

The PID-controllers controlling the propulsion drives are also the same as used in 8.3 and are implemented in the C++ programming language.

8.5.2 Simulation Setup

To compare the two different propulsion configurations a 10 °/10 ° zig-zag test in calm sea is conducted, meaning that the rudder/azimuth angles are given a command of 10 ° and when the heading of the vessel reaches 10 ° the sign of the rudder/azimuth angle commands are changed. As key parameters the surge speed, the heading response and the power consumed by each main thruster are compared. Note that in this zig-zag test the tunnel thruster is not active. Initially, the power plant is started and after 30 s the main propulsors are initiated. The ship is to reach a steady state surge velocity of about 9 kn before the zig-zag manoeuvre is initiated, which happens after about 100 s. The total simulation time is set to 200 s, the global communication time step is set to 10 ms and the integration methods and local time steps for each subsystem is given in Table 8.19. The connections between the subsystems in the co-simulation are summarized in Table 8.20.

8.5.3 Simulation Results

The simulation results comparing the vessel position and heading, the surge speed and the power consumption for the port-side main thruster are shown in Figure 8.25. The lefthand-most plot in the figure shows a north-east-orientation comparison of the two propulsion configurations, the upper righthand-most plot shows a comparison of the surge speed, the second a comparison of vessel heading and the last a comparison of consumed power in a magnified region for the port-side main propulsion unit. As the results indicate, the surge speed is slightly less oscillating

Table 8.20: Connections between subsystems. Note that the connecting variables between subsystems are given in SI-units except for $\mathcal{M}$, $\mathcal{R}$ and τ which are abbreviations for measurement, reference and thrust forces and torques, respectively

CONNECTIONS	**Input**	**Vessel**	$\sum$	**Propulsor**	**Rudder act.**	**El. motor**	**Power plant**	**PID**	**Zig-zag**
Output	⤴	↑	↑	↑	↑	↑	↑	↑	↑
Vessel	→		m/s						$\mathcal{M}$
$\sum$	→	τ		m/s					
Propulsor	→		N, rad			Nm			
Rudder act.	→			rad					
El. motor	→			rad/s			A	$\mathcal{M}$	
Power plant	→					V			
PID	→					$\mathcal{R}$			
Zig-zag	→				$\mathcal{R}$			$\mathcal{R}$	

throughout the manoeuvre for the case including azimuths, as well as the overshooting heading angle and the consumed power are slightly lower for this case in comparison to the conventional propulsion system. The amount of consumed power is also in the expected range, as argued for by the sea trails conducted on Gunnerus that are presented in [213]. To conclude whether one propulsor unit is better than the other for Gunnerus involves more analyses of different scenarios as well as considering more suited models for the power plant and the propulsion drive. However, this is out of scope here.

This case study illustrates the easiness of replacing subsystems in a co-simulation, which is quite interesting when testing different concepts in a fast and virtual setting. This is especially the case when designing new vessels where different vessel equipment or hull designs should be verified to meet the requirement set by the customer in a limited amount of time. This case also illustrates the use of black-box models obtained from different vendors and illustrates clear advantages of using co-simulations when considering different vessel configurations in a design phase of a new-build where different suppliers compete to sell their equipment. Then the equipment can be tested and compared in co-simulations before making a decision as long as the suppliers can provide models of their products.

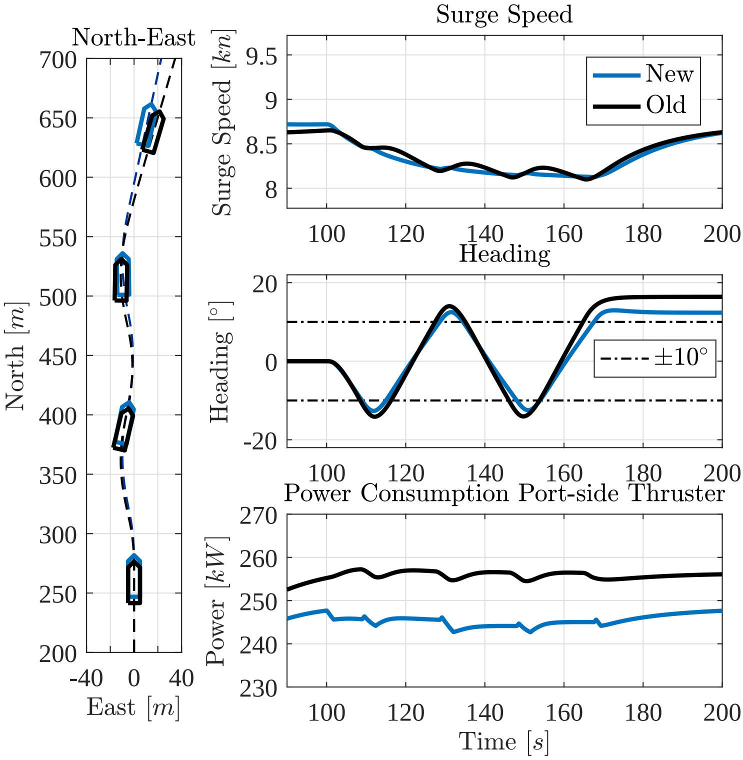


Figure 8.25: Results from zig-zag test comparing the old and the new propulsion systems. Note that *New* denotes the simulation case including azimuth thrusters while *Old* denotes the conventional propulsion system including propellers and rudders

8.6 Conclusion

The main focus in this chapter is to demonstrate the use of co-simulation technology in typical applications obtained from the maritime industry. Four different use-cases and demonstrators have been presented, including collaboration between researchers and different modeling and simulation software, global system optimizing and tuning, the inclusion of hardware in the simulation loop and testing different concepts in a virtual prototyping fashion in an effective and consistent manner. These cases, in addition to the research conducted in the ViProMa project, have brought into light new opportunities in the maritime industry by utilizing co-simulation technology. The use of co-simulations in the maritime industry enables, among others,

1. the use of black-box models which keep secrets related to systems and equipment hidden from competitors. This makes it possible for ship yards to obtain mathematical black-box models of equipment from third party vendors for testing purposes together with the vessel design before determining which equipment to install and the shipyard is able to compare different design concepts before building the vessel.

2. the vessel designer to design the vessel together with the customer on the fly by choosing different concepts from a model library containing many different vessel designs, systems and equipment. It is also expected in the near future that optimization algorithms taking predefined vessel KPIs into consideration can be implemented as a layer on top of the co-simulation platform in order to conduct simulations and choose different equipment based on KPIs from a larger model library.

3. virtual commissioning of vessels and sea trails on beforehand to earlier remove design flaws and implementation errors. It is also expected that the ship yards can demand black-box models from third party vendors in the near future when choosing to buy their equipment. This, in order to be able to test the vessel performance on beforehand, as well as being able to deliver a complete vessel simulator to the customer that can be used for e.g. operation planing and vessel feet optimization, as a *digital twin* of the commissioned vessel. It is also believed that the entire maritime cluster would benefit from working in a *maritime cluster cloud* utilizing co-simulation technology in future research.

However, these topics should be devoted more attention and is way beyond the scope of this thesis, as well as of the ViProMa project.

CHAPTER 9

Conclusions

The work presented in this thesis have been concentrated around the topic of using co-simulations in typical applications relevant for the maritime engineering industry as a new simulation toolbox for supporting research, development and design of new builds and equipment across the entire industry, as was the core-vision in this thesis and the ViProMa project. The topics presented in this thesis have been structured in a chronological order in three different parts. The first part focused on co-simulation theory and closely related challenges, which laid the foundation for developing generic domain models and control systems in the second part. These domain models and control systems were necessary for performing co-simulations of an offshore vessel in DP-operations, as was treated in the last part, also including other case studies. Even though short conclusions are given at the end of each chapter, conclusions on a higher level, linking the work and the main findings more closely together, are given in the following in addition to recommendations for future work.

9.1 Part I: Co-Simulation Theory

The first part, acting as an extended introduction to co-simulation theory, presented and investigated stability theory, accuracy of co-simulation results and tightly coupled systems. As pointed out in Chapter 2 and Chapter 3, co-simulation systems are closely related to sampled systems, being both affected by sampling delays and sampled signal characteristic. These characteristics are caused by discretized system connections and the sampling method, such as ZOH, where the global communication time step size plays a major role. These dynamical sampling effects are thoroughly treated in common sampled system theory, which can be found in the field of control theory, and have therefore not been devoted the most attention in this thesis. However, these sampling effects in combination with numerical solver characteristics, which are essential in co-simulations, bring forth a set of challenges related to stability and accuracy in co-simulations, and bridge dynamical and numerical stability theory, as well as common modeling and control theory. These challenges are related to the fact that each subsystem gets excited by the surrounding subsystems in a co-simulations where the exchanged data between subsystems also contain numerical errors from local solvers which are influenced by the sampling characteristics. Hence, stricter convergence requirements for co-simulation subsystems can be essential for obtaining stable, accurate simulation results. That is also why distributing tightly coupled systems may be a terrible idea because of stiff subsystem couplings with relatively low convergence properties and large potentials for numerical errors, and is also why both the vessel model including a crane in Chapter 5 and the marine power plant in Chapter 6 were implemented as single subsystems.

A method for analysing combined numerical and dynamical stability in co-simulations was derived and investigated in Chapter 3. It should be noted that this method is not presented as a new and practical way of providing stability results for co-simulation systems, on the contrary. As was illustrated in the chapter, analysing the combined stability of a co-simulation system is a demanding task, even if all subsystems only contain linear dynamics, and when using black-box models it is impossible due to the lack of system information. The only reason for presenting this method is to illustrate the complexity of analysing such systems and gives a deeper insight in what affects the stability and accuracy in co-simulation results. This is also why no attempts of analysing the stability of the co-simulation systems presented in Chapter 8 are given. Also, the method is derived based on a combination of system discretization theory and numerical solver theory, and, hence, illustrates the clear correlation between control theory, modeling theory and numerical solver theory in co-simulations. As stated in the chapter, a more practical way of analysing the stability of a co-simulation system is to utilize the subsystem's passivity characteristics, if possible, which is a direct measure of the subsystem's convergence properties.

Another obvious way of increasing the numerical convergence characteristics in a co-simulation is to decrease the global communication time-step size. This is utilized in the ECCO-algorithm presented in section 3.5 which increase the accuracy of the co-simulation results by adaptively controlling the global communication time-step size from estimates of energy residuals from wrongly exchanged energy between connected subsystems. This energy residual is calculated based on comparing the power exchanged between two connected subsystems over time. Note that this energy residual does not necessarily reflect the quantity of energy, only if power bonds between two connected subsystems are considered.

Chapter 4, being the last chapter in Part I, treats tightly coupled systems in co-simulations. Here, the concept of tightly coupled systems is divided into two; systems that are tightly coupled through frequencies and systems that are tightly coupled through causality. The concept of systems that are tightly coupled through frequencies are not defined in continuous system simulations, systems that are not distributed in any sense and solved by the same numerical solver. However, this definition makes more sense in co-simulation systems where fast frequencies in exchanged data can invoke aliasing effects causing instabilities due to high energy residuals. Nevertheless, this type of tightly coupled systems are not given any special attention in this thesis because the only solution is to reduce the global communication time step size such that the Nyquist sampling frequency is not violated, removing possible aliasing effects. On the other hand, systems that are tightly coupled through causality are given the main attention in this chapter.

In general, systems that are tightly coupled through causality are hard to split and distributed because connectivity issues are introduced, as discussed in the chapter. If such connectivity issues exist in a co-simulation system additional functionalities are needed in the co-simulation master algorithm, such as a global implicit numerical solver and re-stepping functionalities, in order to solve the total system. Since this both requires additional work regarding the implementation of additional functionalities in the co-simulation master algorithm and the subsystems, as well as introducing a new source of numerical errors, it is not recommended even though system modularity arguments stand strong. However, a method for reformulating one causality orientation using a low-pass filter with differential effect is presented, investigated and analysed. The method removes possible connectivity issues between two systems that are tightly coupled through causality. Hence, system modularity is restored without requiring additional co-simulation master algorithm functionalities. Also, this method leads to the definition of hybrid causality models – models that have the opportunity of changing causality orientation, and thereby connectivity options, online during a simulation, and is a special class of *switched systems* and related to model switching. In this work hybrid causality models are used for increasing the chance of connecting subsystems and is essential in Chapter 6 where a generic marine power plant model with a weak power grid is derived. Also, hybrid causality models are useful when working with failure conditions in dynamical models, but this topic have not not been in scope in this thesis.

9.2 Part II: Modeling and Control

The second part of the thesis focused on developing generic domain models and control systems and is more or less decoupled from the main focus of co-simulations even though the derived models are intended for co-simulations. Also, each simulation performed in this part of the thesis is a continuous simulation – simulations where the total system is not distributed in any way and solved by only one numerical solver as one self-contained single system.

Chapter 5 derived and presented a generic marine vessel model intended for DP-operations and included a crane placed on the deck of the vessel. The crane and the vessel are tightly coupled systems, tightly coupled through causality, and the main focus in this chapter was to promote the use of Lagrangian dynamics for modeling the vessel and the crane system as one single system, removing any potential differential algebraic loops such that explicit numerical solvers can be used. The chapter also presented lots of subsystems needed for running a simulation of the vessel in DP-operations, such as a DP-control system, a wave-filter, a propulsion system and environmental effects such as irregular sea-states and current. Also, simulations are performed in order to highlight the changes in vessel characteristics when having a small sub-sea payload attached to the crane tip through a wire. Even though the weight of the payload is only about 4 % of the weight of the vessel, it affects the vessel considerably, especially the heading of the vessel and the roll angle. This argue for the use of complex simulators when analysing such vessel operations, even when the payload is small in comparison to the vessel. Two important subsystems which are missing in this vessel model are the power plant and a more sophisticated thrust allocation algorithm that enable rotatable main propulsors, but these are treated in the following two chapters.

A generic marine power plant with weak power grid modeled in the $(d, q, 0)$-reference frame was presented and discussed in Chapter 6 and includes hybrid causality models of the generators, as first presented in Chapter 4 in a case study promoting the use of hybrid causality models. Even though the marine power plant presented in Chapter 4 was simulated as a co-simulation system, the marine power plant is considered as a self-contained single subsystem later on in Chapter 8 because of the collection of small time constants and the necessary small local solver time-step size. Note that the $(d, q, 0)$-reference frame was chosen instead of the (a, b, c)-reference frame in order to avoid subsystems in a co-simulation that are tightly coupled through frequencies, which either introduce aliasing or a low global communication time-step size. Also, the reason for considering hybrid causality generator models was to enable starting and stopping of arbitrary generators without the need of including a small capacitive effect in the power grid, which would have stiffened the total system. In addition to presenting the hybrid causality generator models, the control structure of the total power plant is given much focus. Although all control laws in the total power plant control system are simple PID-based control laws, the total control structure in combination with the PMS functionalities becomes complex and contain much logics.

The last chapter in Part II derived and presented a MPC-based thrust allocation algorithm formulated without including the azimuth thruster angles. In general, many different optimization based thrust allocation algorithms are reported in the literature but most of these formulate the thrust allocation optimization problem by the use of azimuth thruster angles. The reason for removing these angles from the optimization problem is to remove possible singularities and to make the formulation more linear, even though some non-linearities are still present and related to restrictions regarding the angle rates. Nevertheless, the implemented thrust allocation algorithm shows good convergence properties and is benchmarked against a common one-step thrust allocation algorithm. Also, the chapter makes a point out of using the propulsion system to actively filter the oscillatory environmental effects from the DP-control system commands that are not removed by the wave-filter. The results show that the thrust oscillations, and hence, the oscillations in the power consumption, can be reduced significantly without increasing the total power consumption for the propulsion system.

These three chapters in Part II give the building blocks for performing co-simulation case studies in Part III.

9.3 Part III: Applications of Co-Simulations

The last part in the thesis included only one chapter and focused on applications of co-simulations in the maritime industry. Four different case studies were presented and discussed, and included the use of co-simulations as a tool for improving collaborations between researchers, for optimizing system integration, for including hardware in the simulation loop and for testing different vessel configurations in a pre-study before building or upgrading a vessel. Even though these case studies only illustrate a few possible applications for co-simulations, the advantages of using co-simulations as a toolbox in the industry are apparent and related to enabling more complex simulations of multi-domain systems that support the development of safer, greener and more complex marine vessels and equipment. Two major advantages are the use of black-box models for increasing the possibility to cooperate across the industry without spoiling business secrets, and the open co-simulation standard FMI that is supported by various modeling and simulation software, enabling tailored solutions for various systems, software and hardware. The latter was demonstrated in section 8.4 where an Arduino micro-controller was connected to the co-simulation environment using a dedicated communication FMU.

The case studies presented and discussed in Chapter 8 have also contributed to useful experience regarding how to perform co-simulations in practice, and the most significant ones are listed in the following.

9.3.1 Practical Guidelines

During the ViProMa project much experience regarding the application and the practical usage of co-simulations were gained and formulated as practical guidelines by the ViProMa project group. These guidelines are published on the project's web-page [19] and depicted as follows:

Co-Simulations

- Use co-simulation whenever possible to construct full-system models from loosely coupled stand-alone models and modules.
- Be cautious when selecting coupling method and co-simulation communication step size to avoid accuracy and stability issues, see Chapter 3.

Model Interfaces

- Use FMI and package your models as FMUs.
- When selecting the names, types and units of your model variables, decide on a set of conventions and stick to them.
- Document your interfaces well.
- Use standard units of measurement, preferably SI units. If other units are used, explicitly and clearly document so.
- Make use of power bonds to model the flow of energy between subsystems whenever possible, see section 1.3.2.

Stability

- In distributed systems, dynamical system stability can be estimated based on passivity theory [101], through a sink-source analysis of the added- and dissipated energy in the total distributed system. However, note that dynamic stability and numerical stability are closely related in distributed systems through the communication frequency, which affects the eigenvalues in the total system, see Chapter 3.
- The stability in a distributed system should be concluded based on dynamical stability and numerical stability in combination as proposed in section 3.4.2. However, if all subsystems in the distributed system has nice passivity properties, the dynamical- and numerical stability analysis can be separated.

Tightly Coupled Systems

- Systems that are tightly coupled through frequencies and eigenvalues are generally not suited for co-simulation. All the systems that are connected in such tight couplings should be considered implemented as one subsystem in a co-simulation.

- Electrical systems modeled in the (a, b, c)-reference frame are not suited for co-simulation because of the high frequencies in the model interfaces. However, a complete power plant can be implemented as one subsystem and connected to the model environment through mechanical signals such as torque and angular velocity. If single electrical systems are considered to be distributed with "electrical" model interfaces, the $(d, q, 0)$-reference frame, as used in Chapter 4 and Chapter 6, is recommended.
- For systems that are tightly coupled through causality, there are several implementation strategies when considering explicit co-simulation schemes. Three of them are given as follows:

  1. *Correct implementation:* Systems that are tightly coupled through causality are implemented as one subsimulator, as was done for the marine offshore vessel model presented in Chapter 5 including a crane placed at the deck.
  2. *Brute-force implementation:* Negligible dynamical effects can be added to change the causality in one of the subsystems, as discussed in section 4.3.1, although this method often leads to small state offsets and stiff subsystem couplings.
  3. *Hybrid Causality:* A low-pass filter with derivative properties can be used to regain the lost state and to calculate the derivative of the input signal, as shown in section 4.3.2.

- Always avoid distribution of chaotic systems across several subsimulators.

Hybrid Causality

- For subsystems where the preferred causality-connectivity option is hard to determine on beforehand without knowing the connecting environment, the submodel may be implemented as a hybrid causality model to ensure compatibility with other submodels in the distributed simulation environment, see Chapter 4.

9.4 Recommendations for Future Work

The ViProMa project was a knowledge building project and a pilot project for investigating and identifying the need for co-simulation technology in the maritime industry. Even though many different topics were studied in the project, the project's resources were limited. Therefore, the project was only able to briefly scratch the surface in some of the studied topics. Also, the research conducted in the project lay the foundation for further investigations and applications of co-simulations in the maritime industry. A discussion of some recommended topics for future work, structured into different topics, is given in the following.

Stability and Accuracy in Co-Simulations

Stability and accuracy in co-simulations was studied on a higher level in Chapter 3 with the main focus on giving a basic introduction and to provide a baseline for further research such as

- investigating and developing more applicable and conservative methods for analysing stability of co-simulation systems, both numerical stability and dynamical stability. One possible angle may be to look into the total co-simulation system's passivity characteristics in a source-sink approach as mentioned in Chapter 3,
- implementing higher order hold methods in the subsystems in a co-simulation, e.g. FOH as discussed in section 2.2.1, for improving both the stability and accuracy of co-simulation results as well as enable larger co-simulation time step sizes without destabilizing the co-simulation system. This may be achieved by using the FMI 2.0 standard which enables the exchange of signal rates,
- investigating the ECCO method's (see section 3.5) suitability and performance for complex, realistic systems and continue development to improve the method, as well as investigate the use of other algorithms for controlling the accuracy of co-simulation results.

Tightly Coupled Systems

The idea of distributing tightly coupled systems in a co-simulation is in general not recommended, but may be argued for when it comes to modularization of subsystems. Hence, some research topics for further work may include

- investigating the use of hybrid causality models for tightly coupled rigid body systems such that the number of connectivity options is increased and modularity restored,
- investigating the use of implicit global numerical solvers embedded in the co-simulation master algorithm for solving tightly coupled subsystems in a co-simulation. This may be achieved by applying the FMI 2.0 standard.

Model Standardization

To enable seamless use of co-simulation across the maritime industry more work regarding domain model standardization should be initiated. This include

- defining domain model I/O standards, preferably independent on model fidelity, for assuring connectivity of subsystems in co-simulations. This is especially important in the industry, e.g. when multiple potential third party vendors provide black-box models of their products to shipyards for concept evaluation purposes,

- investigating and defining a standard for handling scenarios in co-simulations. This includes a standardization of how environmental effects such as waves, winds and currents, are handled in a co-simulation with multiple subsystems being strongly dependent on these effects, such as the hull and propulsors. Preferably, these environmental effects should be defined as a separate scenario-subsystem which each dependent subsystem have the ability to subscribe to,
- developing more domain models as a framework for fast prototyping and defining key-parameters for these models if possible to facilitate scenario control in co-simulations.

Useful Applications of Co-Simulations

Some areas of application for co-simulations in the maritime industry was illustrated in Chapter 8 and a few demonstrators were developed in the ViProMa project as well as in the VISTA project. However, it is reasonable to believe that co-simulations can be used as a toolbox for solving more complex engineering tasks in the industry such as in

- design optimization where co-simulations are used as a tool for combining and evaluating different vessel concepts in a vessel design phase in a consistent and automated process. This may require further research on combining co-simulations and optimization methods, as well as being able to define different KPIs for evaluating a vessel's performance. One approach is to evaluate different combinations of models in a model library as a discrete optimization approach. However, it is also believed that some domain models can be optimized in co-simulations by adjusting predefined key-parameters,
- surveillance of a vessel in operations where the co-simulation model of the vessel works as a *digital twin* of the vessel and an observer that takes measurements from the actual vessel for both predicting the vessels behaviour in near future and for detecting possible system faults,
- sharing knowledge across the maritime industry, as well as locally across departments in each business, by being a shared co-simulation cluster where all contributors can upload and connect their models to existing ones for evaluating the characteristics of their product in a larger environment in order to optimize and improve their products based on total system performance, not only local subsystem optimization.

References

[1] B. P. Zeigler, H. Praehofer and T. G. Kim. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, 2000.

[2] DNV GL. *Control system network testing and verification*. URL: https://www.dnvgl.com/services/hil-testing-for-marine-systems-82952 (visited on 14/09/2017).

[3] DNV GL. *Software for marine operations - Sesam for marine systems*. URL: https://www.dnvgl.com/services/software-for-marine-operations-sesam-for-marine-systems-2321 (visited on 14/09/2017).

[4] D. Bosich et al. *Thruster motor start-up transient in an all-electric cruise-liner: Numerical simulation and experimental assessment*. 2012 Electrical Systems for Aircraft, Railway and Ship Propulsion. 2012, pp. 1–5.

[5] T. Perez et al. *An Overview of the Marine Systems Simulator (MSS): A Simulink Toolbox for Marine Control Systems*. Modeling, Identification and Control: A Norwegian Research Bulletin 27.4 (2006), pp. 259–275.

[6] T. I. Bo et al. *Marine Vessel and Power Plant System Simulator*. Access, IEEE 3 (2015), pp. 2065–2079.

[7] TU Delft. *ShipDesignLab@TUDelft*. URL: http://marsrv.tudelft.nl/dsdl/ (visited on 14/09/2017).

[8] MIT. *Innovative Ship Design Lab*. URL: http://iship.mit.edu/ (visited on 14/09/2017).

[9] Sintef Ocean. *https://www.sintef.no/en/software/vesim/*. (Visited on 22/09/2017).

[10] A. G. Bruzzone et al. *Simulation-based VV&A methodology for HLA federations: An example from the Aerospace industry*. Proceedings - Simulation Symposium. Vol. 2002-Janua. 2002.

[11] R. D. Lehmer and S. J. Malsom. *Distributed system architecture in VAST-RT for real-time airspace simulation*. Collection of Technical Papers - AIAA Modeling and Simulation Technologies Conference. Vol. 2. 2004.

[12] F. M. Sogandares. *Stone axes and warhammers: A decade of distributed simulation in aviation research*. Proceedings - 16th Workshop on Parallel and Distributed Simulation, PADS 2002. 2002.

[13] R. Skejic and O. M. Faltinsen. *A unified seakeeping and maneuvering analysis of ships in regular waves*. Journal of Marine Science and Technology 13.4 (2008), pp. 371–394.

[14] T. I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. Chichester, UK: John Wiley & Sons Ltd, 2011.

[15] S. Schrödter and T. Gosch. *SESIS - Ship Design and Simulation System*. Proc. COMPIT'08. Liege, 2008.

[16] M. Krammer et al. *Standard compliant co-simulation models for verification of automotive embedded systems*. Forum on Specification and Design Languages. Vol. 2015-Octob. 2015.

[17] M. Winter et al. *From Simulation to Testbench Using the FMI-Standard*. 2015 IEEE Vehicle Power and Propulsion Conference, VPPC 2015 - Proceedings. 2015.

[18] A. Abel et al. *Functional Mock-up Interface in Mechatronic Gearshift Simulation for Commercial Vehicles*. 9th International Modelica Conference. Munich, Germany, 2012, pp. 775–780.

[19] NTNU and Sintef Ocean. *Web-page for the ViProMa-project*. 2017. URL: http://viproma.no (visited on 22/09/2017).

[20] A. J. Sørensen. *A survey of dynamic positioning control systems*. Annual reviews in control 35.1 (2011), pp. 123–136.

[21] V. Hassani et al. *Virtual Prototyping of Maritime Systems and Operations*. Proceedings of the ASME 2016 35th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2016. Busan, South Korea, 2016, pp. 1–7.

[22] SFI Smart Maritime. *https://smartmaritime.no*. 2015. (Visited on 22/09/2017).

[23] S. O. Erikstad et al. *VISTA (Virtual sea trial by simulating complex marine operations): Assessing vessel operability at the design stage*. IMDC 2015 - The International Marine Design Conference. Tokyo, 2015.

[24] L. Meirovitch. *Methods of Analytical Dynamics*. 2003.

[25] D. C. Karnopp, D. L. Margolis and R. C. Rosenberg. *System Dynamics: Modeling and Simulation of Mechatronic Systems*. New York, NY, USA: John Wiley & Sons, Inc., 2006.

[26] W. Borutzky. *Bond graph modelling and simulation of multidisciplinary systems – An introduction*. Simulation Modelling Practice and Theory 17.1 (2009), pp. 3–21.

[27] W. Borutzky. *Bond Graph Modelling of Engineering Systems – Theory, Applications and Software Support*. Ed. by W Borutzky. 1st Editio. NY, NY, U.S.A.: Springer-Verlag, NY, U.S.A, 2010, XVI, 435 p.

[28] S. Skjong and E. Pedersen. *Model-based control designs for offshore hydraulic winch systems*. Ocean Engineering 121 (2016), pp. 224–238.

[29] C.-T. Chen. *Linear System Theory and Design*. 3rd. New York, NY, USA: Oxford University Press, Inc., 1998.

[30] H. K. Khalil. *Nonlinear Systems*. Prentice Hall PTR, 2002.

[31] F. E. Cellier and E. Kofman. *Continuous System Simulation*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[32] E. Hairer, S. P. Nørset and G. Wanner. *Solving Ordinary Differential Equations I*. Springer-Verlag Berlin Heidelberg, 1993.

[33] S. S. Sadjina et al. *Distributed Co-Simulation of Maritime Systems and Operations*. Submitted for publication (2017).

[34] S. Skjong and E. Pedersen. *On the Numerical Stability in Dynamical Distributed Simulations*. Submitted for publication (2017).

[35] S. S. Sadjina et al. *Energy conservation and power bonds in co-simulations: non-iterative adaptive step size control and error estimation*. Engineering with Computers 33.3 (2017), pp. 607–620.

[36] S. Skjong and E. Pedersen. *The Theory of Bond Graphs in Distributed Systems and Simulations*. 2016 International Conference on Bond Graph Modeling and Simulation (ICBGM 2016). Montreal, Canada: Society for Modeling & Simulation International, 2016.

[37] S. Skjong and E. Pedersen. *Hybrid Causality Model Framework including Explicit Reformulation of Differential Algebraic Equations using Perturbations corresponding to added System Dynamics*. Submitted for publication (2017).

[38] B. Rokseth, S. Skjong and E. Pedersen. *Modeling of Generic Offshore Vessel in Crane Operations With Focus on Strong Rigid Body Connections*. IEEE Journal of Oceanic Engineering 42.4 (2017), pp. 846–868.

[39] S. Skjong and E. Pedersen. *A real-time simulator framework for marine power plants with weak power grids*. Elsevier - Journal of Mechatronics 47 (2017), pp. 24–36.

[40] S. Skjong and E. Pedersen. *Non-angular MPC-based Thrust Allocation Algorithm for Marine Vessels - A Study of Optimal Thruster Commands.* IEEE Transactions on Transportation Electrification 3.3 (2017), pp. 792–807.

[41] S. Skjong and E. Pedersen. *Co-Simulation of a Marine Offshore Vessel in DP-Operations including Hardware-In-the-Loop (HIL).* Proceedings of the ASME 2017 36th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2017. Trondheim, Norway, 2017.

[42] S. Skjong et al. *Virtual Prototyping of Maritime Systems and Operations - Applications of Distributed Co-Simulations.* Accepted in Journal of Marine Science and Technology (2017).

[43] K. K. Yum et al. *Simulation of a Hybrid Marine Propulsion System in Waves.* Proceedings of the 28th CIMAC World Congress. Helsinki: CIMAC, 2016.

[44] A. Swider, S. Skjong and E. Pedersen. *Complementary of Data-Driven and Simulation Modeling Based on the Power Plant of the Offshore Vessel.* Proceedings of the ASME 2017 36th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2017. Trondheim, Norway, 2017.

[45] T. Evang, S. Skjong and E. Pedersen. *Crane Rig: An Experimental Setup for Developing and Verifying New Control Methods for Marine Crane Operations.* Proceedings of the ASME 2017 36th International Conference on Ocean, Offshore, and Arctic Engineering OMAE 2017. Trondheim, Norway, 2017.

[46] A. S. Tanenbaum. *Distributed Operating Systems.* Pearson Education, 1995.

[47] A. Tolk. *Engineering Principles of Combat Modeling and Distributed Simulation.* Wiley, 2012.

[48] M. A. Joshi, M. R. Jathar and S. C. Mehrotra. *Distributed System for Weather Data Collection through TINI Microcontroller.* International Journal of Environmental Science and Development 2.1 (2011), pp. 70–72.

[49] A. S. Tanenbaum and M. Van Steen. *Distributed Systems: Principles and Paradigms.* Pearson Prentice Hall, 2007.

[50] S. Ghosh. *Distributed Systems: An Algorithmic Approach, Second Edition.* Chapman & Hall/CRC Computer and Information Science Series. CRC Press, 2014.

[51] C. Gomes et al. *Co-simulation: State of the art.* CoRR abs/1702.0 (2017). URL: http://arxiv.org/abs/1702.00686.

[52] G. F. Coulouris, J. Dollimore and T. Kindberg. *Distributed systems: concepts and design*. pearson education, 2005.

[53] H. Parthasarathy. *Textbook of Signals and Systems*. I.K. International Publishing House Pvt. Limited, 2004.

[54] N. F. Macia and G. J. Thaler. *Modeling and Control of Dynamic Systems*. Thomson Delmar Learning, 2005.

[55] O. W. Eshbach, B. D. Tapley and T. R. Poston. *Eshbach's Handbook of Engineering Fundamentals*. A Wiley-Interscience publication. Wiley, 1990.

[56] Z. Zhang and K. T. Chong. *Comparison between first-order hold with zero-order hold in discretization of input-delay nonlinear systems*. 2007 International Conference on Control, Automation and Systems. 2007, pp. 2892–2896.

[57] K. M. Passino and P. J. Antsaklis. *Inverse stable sampled low-pass systems*. International Journal of Control 47.6 (1988), pp. 1905–1913.

[58] L. A. Hageman. *Applied Iterative Methods*. Elsevier Science, 2016.

[59] J. S. Dahmann, R. M. Fujimoto and R. M. Weatherly. *The Department of Defense High Level Architecture*. Proceedings of the 1997 Winter Simulation Conference (1997), pp. 142–149.

[60] T. Blochwitz et al. *The Functional Mockup Interface for Tool independent Exchange of Simulation Models*. 8th International Modelica Conference 2011 (2009), pp. 173–184.

[61] A. Garro and A. Falcone. *On the Integration of HLA and FMI for Supporting Interoperability and Reusability in Distributed Simulation*. Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium. DEVS '15. San Diego, CA, USA: Society for Computer Simulation International, 2015, pp. 9–16.

[62] M. U. Awais et al. *Using the HLA for Distributed Continuous Simulations*. 2013 8th EUROSIM Congress on Modelling and Simulation. 2013, pp. 544–549.

[63] M. U. Awais et al. *Distributed hybrid simulation using the HLA and the Functional Mock-up Interface*. IECON Proceedings (Industrial Electronics Conference). 2013, pp. 7564–7569.

[64] J. S. Dahmann. *High Level Architecture for simulation*. Proceedings First International Workshop on Distributed Interactive Simulation and Real Time Applications. 1997, pp. 9–14.

[65] Simulation Interoperability Standards Organization. *IEEE standard for modeling and simulation (M&S) high level architecture (HLA) framework and rules*. IEEE Std. 1516-2010. 2010.

[66] IEEE. *IEEE Standard for Distributed Interactive Simulation–Application Protocols*. IEEE Std 1278.1-2012 (Revision of IEEE Std 1278.1-1995) (2012), pp. 1–747.

[67] N. Husteli. *Common simulation interface. Documentation and Tutorial*. Tech. rep. Trondheim: MARINTEK, Norway, 2005.

[68] T. Blochwitz et al. *Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models*. Proceedings of the 9th International Modelica Conference. The Modelica Association, 2012, pp. 173–184.

[69] Official webpage for the FMI standard. *https://fmi-standard.org*. (Visited on 22/09/2017).

[70] NTNU's research vessel R/V Gunnerus. *https://www.ntnu.edu/oceans/gunnerus*. (Visited on 22/09/2017).

[71] Controllab. *http://www.20sim.com/*. (Visited on 22/09/2017).

[72] M. Arnold. *Numerical methods for simulation in applied dynamics*. Simulation Techniques for Applied Dynamics. Ed. by M Arnold and W Schiehlen. Vol. 507. CISM International Centre for Mechanical Sciences. Springer Vienna, 2009. Chap. 5, pp. 191–246.

[73] T. Chen and B. A. Francis. *Input-Output Stability Of Sampled-Data Systems*. IEEE Transactions on Automatic Control 36.1 (1991), pp. 50–58.

[74] T. Chen and B. A. Francis. *H2-Optimal Sampled-Data Control*. IEEE Transactions on Automatic Control 36.4 (1991), pp. 387–397.

[75] S. Gottlieb, C.-W. Shu and E. Tadmor. *Strong stability-preserving high-order time discretization methods*. SIAM Review 43.1 (2001), pp. 89–112.

[76] R. Kübler and W. Schiehlen. *Two Methods of Simulator Coupling*. Mathematical and Computer Modelling of Dynamical Systems 6.2 (2000), pp. 93–113.

[77] M. Arnold, C. Clauss and T. Schierz. *Error Analysis and Error Estimates for Co-Simulation in FMI for Model Exchange and Co-Simulation V2.0*. Archive of Mechanical Engineering 60.1 (2013), pp. 75–94.

[78] M. Arnold. *Stability of sequential modular time integration methods for coupled multibody system models*. Journal of Computational and Nonlinear Dynamics 5.3 (2010), pp. 1–9.

[79] B. Owren. *Stability of Runge-Kutta methods used in modular integration*. Journal of Computational and Applied Mathematics 62.1 (1995), pp. 89–101.

[80] M. Busch. *Zur effizienten Kopplung von Simulationsprogrammen*. Thesis. 2012.

[81] G. Dahlquist. *Convergence and stability in the numerical integration of ordinary differential equations*. Mathematica Scandinavica 4 (1956), pp. 33–53.

[82] B. Schweizer, P. Li and D. Lu. *Explicit and Implicit Cosimulation Methods: Stability and Convergence Analysis for Different Solver Coupling Approaches*. Journal of Computational and Nonlinear Dynamics 10.5 (2015), p. 51007.

[83] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. 1998, p. 332.

[84] R. Kalman and J. Bertram. *Control system analysis and design via the second method of lyapunov: (I) continuous-time systems (II) discrete time systems*. IRE Transactions on Automatic Control 4.3 (1959), p. 112.

[85] S. Rudra, R. K. Barai and M. Maitra. *Block Backstepping Design of Nonlinear State Feedback Control Law for Underactuated Mechanical Systems*. Springer Singapore, 2016.

[86] G. Bartolini et al. *Modern Sliding Mode Control Theory: New Perspectives and Applications*. Lecture Notes in Control and Information Sciences. Springer Berlin Heidelberg, 2008.

[87] B. Wilamowski and J. D. Irwin. *Control and Mechatronics*. ENGnetBASE 2015. CRC Press, 2016.

[88] R. H. Gielen. *Stability analysis and control of discrete-time systems with delay*. Doctorial Thesis. Technische Universiteit Eindhoven, 2013, p. 169.

[89] I. Karafyllis and C. Kravaris. *Global stability results for systems under sampled-data control*. 2007 European Control Conference (ECC). 2007, pp. 5761–5768.

[90] R. Goebel and R. G. Sanfelice. *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*. Princeton University Press, 2012.

[91] R. Goebel, R. G. Sanfelice and A. R. Teel. *Hybrid dynamical systems. Robust stability and control for systems that combine continuous-time and discrete-time dynamics*. IEEE Control Systems Magazine 29.2 (2009), pp. 28–93.

[92] H. J. Stetter. *Analysis of Discretization Methods for Ordinary Differential Equations*. Springer Tracts in Natural Philosophy. Springer Berlin Heidelberg, 2013.

[93] R. E. Kalman and J. E. Bertram. *Control System Analysis and Design Via the "Second Method" of Lyapunov: II—Discrete-Time Systems*. Journal of Basic Engineering 82.2 (1960), pp. 394–400.

[94] R. E. Kalman. *Nonlinear aspects of sampled-data control systems*. in Proceedings of Symposium on Nonlinear Circuit Analysis. Ed. by Polytechnic Institute of Brooklyn edited by J. Fox. 1956, pp. 273–313.

[95] L. Grune and D. Nesic. *Stabilization of sampled-data nonlinear systems via their approximate models: an optimization based approach*. Proceedings of the 41st IEEE Conference on Decision and Control, 2002. Vol. 2. 2002, 1934–1939 vol.2.

[96] D. Nesic and A. R. Teel. *A framework for stabilization of nonlinear sampled-data systems based on their approximate discrete-time models*. IEEE Transactions on Automatic Control 49.7 (2004), pp. 1103–1122.

[97] E. D. Sontag. *Input to State Stability: Basic Concepts and Results*. Nonlinear and Optimal Control Theory: Lectures given at the C.I.M.E. Summer School held in Cetraro, Italy June 19–29, 2004. Ed. by Paolo Nistri and Gianna Stefani. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 163–220.

[98] M. Arcak, C. Meissen and A. Packard. *Networks of Dissipative Systems*. SpringerBriefs in Electrical and Computer Engineering. Cham: Springer International Publishing, 2016.

[99] N. Kottenstette and P. J. Antsaklis. *Relationships between Positive Real, Passive Dissipative, & Positive Systems*. 2010.

[100] A. J. der Schaft. *L2-Gain Analysis of Nonlinear Systems and Nonlinear State Feedback H-infty Control*. IEEE Transactions on Automatic Control 37.6 (1992), pp. 770–784.

[101] H. K. Khalil. *Nonlinear Systems, Third Edition*. 2002, p. 750.

[102] E. M. Navarro-López. *Several dissipativity and passivity implications in the linear discrete-time setting*. Mathematical Problems in Engineering 2005.6 (2005), pp. 599–616.

[103] J. C. Willems. *Dissipative dynamical systems part I: General theory*. Archive for Rational Mechanics and Analysis 45.5 (1972), pp. 321–351.

[104] J. C. Willems. *Dissipative dynamical systems Part II: Linear systems with quadratic supply rates*. Archive for Rational Mechanics and Analysis 45.5 (1972), pp. 352–393.

[105] E. M. Navarro-López and E. Fossas-Colet. *Feedback passivity of nonlinear discrete-time systems with direct input–output link*. Automatica 40.8 (2004), pp. 1423–1428.

[106] S. Monaco and D. Normand-Cyrot. *On the conditions of passivity and losslessness in discrete time*. 1997 European Control Conference (ECC). 1997, pp. 3621–3625.

[107] C. I. Byrnes and W. Lin. *Discrete-time lossless systems, feedback equivalence and passivity*. Proceedings of 32nd IEEE Conference on Decision and Control. 1993, 1775–1781 vol.2.

[108] S. Monaco, D. Normand-Cyrot and F. Tiefensee. *From passivity under sampling to a new discrete-time passivity concept*. 2008 47th IEEE Conference on Decision and Control. 2008, pp. 3157–3162.

[109] P. Naghshtabrizi, J. P. Hespanha and A. R. Teel. *On the robust stability and stabilization of sampled-data systems: A hybrid system approach*. Proceedings of the IEEE Conference on Decision and Control. 2006, pp. 4873–4878.

[110] M. Busch and B. Schweizer. *An explicit approach for controlling the macro-step size of co-simulation methods*. Proceedings of the 7th European Nonlinear Dynamics Conference (ENOC 2011): July 24 - 29, 2011, Rome, Italy. Ed. by D Bernadini. 2011, pp. 1–6.

[111] D. Sahm. *A two-axis, bond graph model of the dynamics of synchronous electrical machines*. Journal of the Franklin Institute 308.3 (1979), pp. 205–218.

[112] D. Karnopp. *An approach to derivative causality in bond graph models of mechanical systems*. Journal of the Franklin Institute 329.1 (1992), pp. 65–75.

[113] R. C. Rosenberg. *State-space formulation for bond graph models of multiport systems*. Journal of Dynamic Systems, Measurement, and Control 93.1 (1971), pp. 35–40.

[114] E. Hairer, S. P. Norsett and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer, 2010.

[115] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2010.

[116] J. C. Butcher. *Implicit runge-kutta processes*. Mathematics of Computation 18.85 (1964), pp. 50–64.

[117] L. Tavernini. *Differential automata and their discrete simulators*. Nonlinear Analysis: Theory, Methods & Applications 11.6 (1987), pp. 665–683.

[118] A. Back, J. Guckenheimer and M. Myers. *A dynamical simulation facility for hybrid systems*. Hybrid Systems. Ed. by Robert L Grossman et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 255–267.

[119] A. Nerode and W. Kohn. *Models for hybrid systems: Automata, topologies, controllability, observability*. Hybrid Systems. Ed. by Robert L Grossman et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993, pp. 317–356.

[120] R. W. Brockett. *Hybrid Models for Motion Control Systems*. Essays on Control: Perspectives in the Theory and its Applications. Ed. by H L Trentelman and J C Willems. Boston, MA: Birkh{ä}user Boston, 1993, pp. 29–53.

[121] M. S. Branicky. *Universal computation and other capabilities of hybrid and continuous dynamical systems*. Theoretical Computer Science 138.1 (1995), pp. 67–100.

[122] K. Edström. *Switched Bond Graphs: Simulation and Analysis*. PhD thesis. Linköping University, Automatic Control, 1999, p. 290.

[123] A. C. Umarikar and L. Umanand. *Modelling of switching systems in bond graphs using the concept of switched power junctions*. Journal of the Franklin Institute 342.2 (2005), pp. 131–147.

[124] F. T. Brown. *Hamiltonian and Lagrangian bond graphs*. Journal of the Franklin Institute 328.5-6 (1991), pp. 809–831.

[125] S. Skjong. *Master Thesis: Modeling , Simulation and Control of Hydraulic Winch System*. June. Trondheim: NTNU, 2014.

[126] *MSS Marine systems simulator*. http://www.marinecontrol.org/.

[127] T. Perez et al. *An Overview of the Marine Systems Simulator (MSS): A Simulink Toolbox for Marine Control Systems*. Modeling, Identification and Control 27.4 (2006), pp. 259–275.

[128] T. I. Bø et al. *Real-Time Marine Vessel and Power Plant Simulation*. ASME 2015 34th International Conference on Ocean, Offshore and Arctic Engineering. American Society of Mechanical Engineers. 2015, V001T01A007–V001T01A007.

[129] T. I. Bø. *Scenario- and Optimization-based Control of Marine Electric Power Systems*. PhD thesis. Norwegian University of Science and Technology (NTNU), 2016.

[130] T. I. Fossen. *Marine Control Systems: Navigation and Control of Ships, Rigs and Underwater Vehicles*. Marine Cybernetics, Trondheim, 2002.

[131] A. J. Sørensen, E. Pedersen and Ø. N. Smogeli. *Simulation-based design and testing of dynadynamic positioned marine vessels*. International Conference on Marine Simulation and Ship Maneuverability. 2004.

[132] T. Perez and M. Blanke. *DCMV a Matlab/Simulink Toolbox for Dynamics and Control of Marine Vehicles*. Proceedings 6th Conference on Manoeuvering and Control of Marine Craft, MCMC. 2003.

[133] W. Borutzky. *Bond graph modeling from an object oriented modeling point of view*. Simulation Practice and Theory 7.5-6 (1999), pp. 439–461.

[134] P. J. From, K. Y. Pettersen and J.T. Gravdahl. *Singularity-free dynamic equations of AUV-manipulator systems*. Proceeding of symposium on intelligent autonomous vehicles. 2010.

[135] J. Xu and K. H. Halse. *A generic modelling approach for heavy lifting marine operations*. Sun Above the Horizon: Meteoric Rise of the Solar Industry 5 (2014), p. 89.

[136] F. Sanfilippo et al. *Flexible Modeling And Simulation Architecture For Haptic Control Of Maritime Cranes And Robotic Arm*. ECMS. 2013, pp. 235–242.

[137] B. Rokseth. *Dynamic Model and Serial Communication on 7 Degrees-of-freedom Manipulator*. Tech. rep. 2013.

[138] B. Rokseth and E. Pedersen. *A Bond Graph Approach for Modelling of Robot Manipulators*. International Conference on Bond Graph Modeling and Simulation. 2014, pp. 728–734.

[139] J. H. Ginsberg. *Advanced Engineering Dynamics*. 2nd ed. 1995.

[140] Controllab Product. *20-sim Reference 4.3*. 2013.

[141] S. Birkett, J. Thoma and P. Roe. *A pedagogical analysis of bond graph and linear graph physical system models*. Mathematical and Computer Modelling of Dynamical Systems 12.2-3 (2006), pp. 107–125.

[142] O. Faltinsen. *Sea Loads on Ships and Offshore Structures*. Cambridge Ocean Technology Series. Cambridge University Press, 1993.

[143] J. N. Newman. *Marine Hydrodynamics*. MIT, Cambridge, 1977.

[144] D. R. Yoerger, J. G. Cooke and J.-J. E. Slotine. *The influence of thruster dynamics on underwater vehicle behavior and their incorporation into control system design*. IEEE Journal of Oceanic Engineering 15.3 (1990), pp. 167–178.

[145] J. Kim and W. K. Chung. *Accurate and practical thruster modeling for underwater vehicles*. Ocean Engineering 33.5–6 (2006), pp. 566–586.

[146] A. J. Healey et al. *Toward an improved understanding of thruster dynamics for underwater vehicles*. Proceedings of IEEE Symposium on Autonomous Underwater Vehicle Technology (AUV'94). Vol. 20. 4. IEEE, 1995, pp. 340–352.

[147] T. I. Fossen and T. A. Johansen. *A survey of control allocation methods for ships and underwater vehicles*. Control and Automation, 2006. MED'06. 14th Mediterranean Conference on. IEEE. 2006, pp. 1–6.

[148] T. I. Fossen and J. P. Strand. *Passive nonlinear observer design for ships using lyapunov methods: full-scale experiments with a supply vessel*. Automatica 35.1 (1999), pp. 3–16.

[149] S. Skjong and E. Pedersen. *Modeling hydraulic winch system*. 2014 International Conference on Bond Graph Modeling and Simulation (ICBGM 2014). Vol. 46. 8. Monterey, CA: The Society for Modeling and Simulation International, 2014, pp. 181–187.

[150] L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators*. 1st ed. 2000.

[151] R. Nilsen and I. Sorfonn. *Hybrid Power Generation Systems*. 2009 13th European Conference on Power Electronics and Applications. 2009, pp. 1–9.

[152] E. K. Dedes, D. A. Hudson and S. R. Turnock. *Assessing the potential of hybrid energy technology to reduce exhaust emissions from global shipping*. Energy Policy 40 (2012), pp. 204–218.

[153] M. Morsy El-Gohary. *Overview of past, present and future marine power plants*. Journal of Marine Science and Application 12.2 (2013), pp. 219–227.

[154] T. A. Pedersen and E. Pedersen. *Bond graph modelling of marine power systems*. Mathematical and Computer Modelling of Dynamical Systems 18.2 (2012), pp. 153–173.

[155] J. Machowski, J. W. Bialek and J. R. Bumby. *Power system dynamics: stability and control*. 2nd. Chichester, U.K. Wiley, 2008.

[156] A. Davoudi, L. Qu and P. L. Chapman. *Summary of Recent Work on Reduction Techniques Applied to Electromechanical Modeling*. 2007 IEEE Electric Ship Technologies Symposium. IEEE, 2007, pp. 363–370.

[157] A. Davoudi. *Reduced-order Modeling of Power Electronics Components and Systems*. University of Illinois, 2010.

[158] J. Cathey, R. Cavin and A. Ayoub. *Transient Load Model of an Induction Motor*. IEEE Transactions on Power Apparatus and Systems PAS-92.4 (1973), pp. 1399–1406.

[159] D. P. Carroll and P. C. Krause. *Stability Analysis of a DC Power System*. IEEE Transactions on Power Apparatus and Systems PAS-89.6 (1970), pp. 1112–1119.

[160] C. Cho et al. *Active Synchronizing Control of a Microgrid*. IEEE Transactions on Power Electronics 26.12 (2011), pp. 3707–3719.

[161] M. N. Marwali, J.-W. Jung and A. Keyhani. *Control of distributed generation systems - Part II: Load sharing control*. IEEE Transactions on Power Electronics 19.6 (2004), pp. 1551–1561.

[162] P. J. Mosterman and G. Biswas. *A theory of discontinuities in physical system models*. Journal of the Franklin Institute 335.3 (1998), pp. 401–439.

[163] C. D. Beers et al. *Building Efficient Simulations from Hybrid Bond Graph Models*. IFAC Proceedings Volumes 39.5 (2006), pp. 71–76.

[164] T. A. Pedersen. *Bond Graph Modeling of Marine Power Systems*. PhD thesis. Norwegian University of Science and Technology, Department of Marine Technology, 2009.

[165] P. C. Krause and C. H. Thomas. *Simulation of Symmetrical Induction Machinery*. IEEE Transactions on Power Apparatus and Systems 84.11 (1965), pp. 1038–1053.

[166] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Statewide Agricultural Land Use Baseline 2015 1 (1998), p. 332.

[167] J. Belanger, P. Venne and J.-N. Paquin. *The What, Where and Why of Real-Time Simulation*. Planet RT 1.0 (2010), pp. 37–49.

[168] J. B. Heywood. *Internal Combustion Engine Fundamentals*. Automotive technology series. McGraw-Hill, 1988.

[169] J. P. Louis. *Control of Synchronous Motors*. ISTE. Wiley, 2013.

[170] S. Funabiki et al. *Automatic voltage regulation of synchronous generator with pole assignment self-tuning regulator*. Industrial Electronics, Control and Instrumentation, 1991. Proceedings. IECON ’91., 1991 International Conference on. 1991, 1807–1811 vol.3.

[171] K. J. Astrom and L. Rundqwist. *Integrator Windup and How to Avoid It*. 1989 American Control Conference. 1989, pp. 1693–1698.

[172] Y. Han et al. *Review of Active and Reactive Power Sharing Strategies in Hierarchical Controlled Microgrids*. IEEE Transactions on Power Electronics 32.3 (2017), pp. 2427–2451.

[173] R. E. Cosse et al. *Turbine/generator governor droop/isochronous fundamentals - A graphical approach*. 2011 Record of Conference Papers Industry Applications Society 58th Annual IEEE Petroleum and Chemical Industry Conference (PCIC). 2011, pp. 1–8.

[174] D. J. McGowan, D. J. Morrow and B. Fox. *Integrated governor control for a diesel-generating set*. IEEE Transactions on Energy Conversion 21.2 (2006), pp. 476–483.

[175] T. A. Johansen and T. I. Fossen. *Control allocation - A survey*. Automatica 49.5 (2013), pp. 1087–1103.

[176] M. V. Kothare, V. Balakrishnan and M. Morari. *Robust constrained model predictive control using linear matrix inequalities*. Automatica 32.10 (1996), pp. 1361–1379.

[177] C. E. Garcia, D. M. Prett and M. Morari. *Model predictive control: Theory and practice-A survey*. Automatica 25.3 (1989), pp. 335–348.

[178] D. Q. Mayne. *Model predictive control: Recent developments and future promise*. Automatica 50.12 (2014), pp. 2967–2986.

[179] J. M. Maciejowski. *Predictive control : with constraints*. Prentice Hall; 1 edition (September 6, 2000), 2002, p. 331.

[180] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*. Communications and Control Engineering. London: Springer London, 2011.

[181] A. Veksler et al. *Thrust Allocation With Dynamic Power Consumption Modulation for Diesel-Electric Ships*. IEEE Transactions on Control Systems Technology 24.2 (2015), pp. 578–593.

[182] S. P. Berge and T. I. Fossen. *Robust control allocation of overactuated ships; experiments with a model ship*. Proc. 4th IFAC Conf. Manoeuvring and Control of Marine Craft. 1997, pp. 166–171.

[183] M. Rindarøy and T. A. Johansen. *Fuel Optimal Thrust Allocation in Dynamic Positioning*. IFAC Proceedings Volumes 46.33 (2013), pp. 43–48.

[184] A. Veksler, T. A. Johansen and R. Skjetne. *Thrust allocation with power management functionality on dynamically positioned vessels*. 2012 American Control Conference (ACC). IEEE, 2012, pp. 1468–1475.

[185] W. Durham. *Constrained control allocation*. Journal of Guidance, Control, and Dynamics 16.4 (1993), pp. 717–725.

[186] W. Durham. *Efficient, Near-Optimal Control Allocation*. Journal of Guidance, Control, and Dynamics 22.2 (1999), pp. 369–372.

[187] T. I. Fossen. *Guidance and Control of Ocean Vehicles*. Chichester, United Kingdom: John Wiley and Sons Ltd, 1994, p. 494.

[188] T. A. Johansen, I. Petersen and O. Slupphaug. *Explicit sub-optimal linear quadratic regulation with state and input constraints*. Automatica 38.7 (2002), pp. 1099–1111.

[189] P. Tøndel, T. A. Johansen and A. Bemporad. *An algorithm for multi-parametric quadratic programming and explicit MPC solutions*. Automatica 39.3 (2003), pp. 489–497.

[190] T. A. Johansen, T. I. Fossen and S. P. Berge. *Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming*. Control Systems Technology, IEEE Transactions on 12.1 (2004), pp. 211–216.

[191] P. Tøndel, T. A. Johansen and A. Bemporad. *Evaluation of piecewise affine control via binary search tree*. Automatica 39.5 (2003), pp. 945–950.

[192] A. Veksler et al. *Dynamic Positioning With Model Predictive Control*. IEEE Transactions on Control Systems Technology 24.4 (2016), pp. 1340–1353.

[193] A. J. Sørensen, S. I. Sagatun and T. I. Fossen. *Design of a dynamic positioning system using model-based control*. Control Engineering Practice 4.3 (1996), pp. 359–368.

[194] P. T. K. Fung and M. J. Grimble. *Dynamic Ship Positioning Using a Self-Tuning Kalman Filter*. IEEE Transactions on Automatic Control 28.3 (1983), pp. 339–350.

[195] S. Saelid, N. Jenssen and J. Balchen. *Design and analysis of a dynamic positioning system based on Kalman filtering and optimal control*. IEEE Transactions on Automatic Control 28.3 (1983), pp. 331–339.

[196] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering*. 4th. J. Wiley, 1992.

[197] A. Richards. *Fast Model Predictive Control with soft constraints*. European Journal of Control 25 (2015), pp. 51–59.

[198] B. Houska, H. J. Ferreau and M. Diehl. *ACADO toolkit-An open-source framework for automatic control and dynamic optimization*. Optimal Control Applications and Methods 32.3 (2011), pp. 298–312.

[199] H. J. Ferreau et al. *qpOASES: a parametric active-set algorithm for quadratic programming*. Mathematical Programming Computation 6.4 (2014), pp. 327–363.

[200] L. Pivano, Ø. N. Smogeli and B. Vik. *DynCap - The Next Level Dynamic DP Capability Analysis*. Proceedings of the 2nd Marine Operations Specialty Symposium. Singapore: Research Publishing Services, 2012, pp. 17–26.

[201] J. H. Evans. *Basic Design Concepts*. Journal of the American Society for Naval Engineers 71.4 (1959), pp. 671–678.

[202] T. Ulstein and P. O. Brett. *What is a better ship?–It all depends…* 12th International Marine Design Conference. Vol. 1. 2015.

[203] K. K. Yum and E. Pedersen. *Architecture of model libraries for modelling turbocharged diesel engines*. Mathematical and Computer Modelling of Dynamical Systems 22.6 (2016), pp. 584–612.

[204] K. K. Yum et al. *Simulation of a two-stroke diesel engine for propulsion in waves*. International Journal of Naval Architecture and Ocean Engineering 9.4 (2017), pp. 351–372.

[205] M. Hepperle. *JavaFoil*. Web Page. URL: http://www.mh-aerotools.de/ (visited on 22/09/2017).

[206] O. M. Faltinsen et al. *Prediction of resistance and propulsion of a ship in a seaway*. Conference Paper. 1980.

[207] K. Minsaas, O. M. Faltinsen and B. Persson. *On the importance of added resistance, propeller immersion and propeller ventilarion for large ships in a seaway*. 2nd International Symposium on Practical Design in Shipbuilding. Tokyo and Seoul, 1983, pp. 149–159.

[208] B. Taskar and S. Steen. *Analysis of Propulsion Performance of KVLCC2 in Waves*. Fourth International Symposium on Marine Propulsors, Austin, Texas, USA. 2015.

[209] SIMMAN. *MOERI Tanker KVLCC2*. 2008. URL: http://www.simman2008.dk/KVLCC/KVLCC2/tanker2.html (visited on 22/09/2017).

[210] G. K. Singh et al. *A simple indirect field-oriented control scheme for multiconverter-fed induction motor*. IEEE Transactions on Industrial Electronics 52.6 (2005), pp. 1653–1659.

[211] Arduino. *Company web page*. URL: https://www.arduino.org/ (visited on 22/09/2017).

[212] A. Ross et al. *Identification of Nonlinear Manoeuvring Models for Marine Vessels using planar motion mechanism tests*. 34th International Conference on Ocean, Offshore and Arctic Engineering (OMAE2015), St. John's, Newfoundland, Canada, 2015.

[213] Ø. Selvik, T. Berg and S. Gavrilin. *Sea trials for validation of shiphandling simulation models – a case study*. Maritime-Port Technology and Development. CRC Press, 2014, pp. 141–146.

APPENDIX A

Previous PhD theses published at the Departement of Marine Technology

Previous PhD theses published at the Departement of Marine Technology (earlier: Faculty of Marine Technology) NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Report No.	Author	Title
	Kavlie, Dag	Optimization of Plane Elastic Grillages, 1967
	Hansen, Hans R.	Man-Machine Communication and Data-Storage Methods in Ship Structural Design, 1971
	Gisvold, Kaare M.	A Method for non-linear mixed -integer programming and its Application to Design Problems, 1971
	Lund, Sverre	Tanker Frame Optimalization by means of SUMT-Transformation and Behaviour Models, 1971
	Vinje, Tor	On Vibration of Spherical Shells Interacting with Fluid, 1972
	Lorentz, Jan D.	Tank Arrangement for Crude Oil Carriers in Accordance with the new Anti-Pollution Regulations, 1975
	Carlsen, Carl A.	Computer-Aided Design of Tanker Structures, 1975
	Larsen, Carl M.	Static and Dynamic Analysis of Offshore Pipelines during Installation, 1976
UR-79-01	Brigt Hatlestad, MK	The finite element method used in a fatigue evaluation of fixed offshore platforms. (Dr.Ing. Thesis)
UR-79-02	Erik Pettersen, MK	Analysis and design of cellular structures. (Dr.Ing. Thesis)
UR-79-03	Sverre Valsgård, MK	Finite difference and finite element methods applied to nonlinear analysis of plated structures. (Dr.Ing. Thesis)
UR-79-04	Nils T. Nordsve, MK	Finite element collapse analysis of structural members considering imperfections and stresses due to fabrication. (Dr.Ing. Thesis)
UR-79-05	Ivar J. Fylling, MK	Analysis of towline forces in ocean towing systems. (Dr.Ing. Thesis)
UR-80-06	Nils Sandsmark, MM	Analysis of Stationary and Transient Heat Conduction by the Use of the Finite Element Method. (Dr.Ing. Thesis)
UR-80-09	Sverre Haver, MK	Analysis of uncertainties related to the stochastic modeling of ocean waves. (Dr.Ing. Thesis)
UR-81-15	Odland, Jonas	On the Strength of welded Ring stiffened cylindrical Shells primarily subjected to axial Compression
UR-82-17	Engesvik, Knut	Analysis of Uncertainties in the fatigue Capacity of

		Welded Joints
UR-82-18	Rye, Henrik	Ocean wave groups
UR-83-30	Eide, Oddvar Inge	On Cumulative Fatigue Damage in Steel Welded Joints
UR-83-33	Mo, Olav	Stochastic Time Domain Analysis of Slender Offshore Structures
UR-83-34	Amdahl, Jørgen	Energy absorption in Ship-platform impacts
UR-84-37	Mørch, Morten	Motions and mooring forces of semi submersibles as determined by full-scale measurements and theoretical analysis
UR-84-38	Soares, C. Guedes	Probabilistic models for load effects in ship structures
UR-84-39	Aarsnes, Jan V.	Current forces on ships
UR-84-40	Czujko, Jerzy	Collapse Analysis of Plates subjected to Biaxial Compression and Lateral Load
UR-85-46	Alf G. Engseth, MK	Finite element collapse analysis of tubular steel offshore structures. (Dr.Ing. Thesis)
UR-86-47	Dengody Sheshappa, MP	A Computer Design Model for Optimizing Fishing Vessel Designs Based on Techno-Economic Analysis. (Dr.Ing. Thesis)
UR-86-48	Vidar Aanesland, MH	A Theoretical and Numerical Study of Ship Wave Resistance. (Dr.Ing. Thesis)
UR-86-49	Heinz-Joachim Wessel, MK	Fracture Mechanics Analysis of Crack Growth in Plate Girders. (Dr.Ing. Thesis)
UR-86-50	Jon Taby, MK	Ultimate and Post-ultimate Strength of Dented Tubular Members. (Dr.Ing. Thesis)
UR-86-51	Walter Lian, MH	A Numerical Study of Two-Dimensional Separated Flow Past Bluff Bodies at Moderate KC-Numbers. (Dr.Ing. Thesis)
UR-86-52	Bjørn Sortland, MH	Force Measurements in Oscillating Flow on Ship Sections and Circular Cylinders in a U-Tube Water Tank. (Dr.Ing. Thesis)
UR-86-53	Kurt Strand, MM	A System Dynamic Approach to One-dimensional Fluid Flow. (Dr.Ing. Thesis)
UR-86-54	Arne Edvin Løken, MH	Three Dimensional Second Order Hydrodynamic Effects on Ocean Structures in Waves. (Dr.Ing. Thesis)
UR-86-55	Sigurd Falch, MH	A Numerical Study of Slamming of Two-Dimensional Bodies. (Dr.Ing. Thesis)
UR-87-56	Arne Braathen, MH	Application of a Vortex Tracking Method to the Prediction of Roll Damping of a Two-Dimension Floating Body. (Dr.Ing. Thesis)

UR-87-57	Bernt Leira, MK	Gaussian Vector Processes for Reliability Analysis involving Wave-Induced Load Effects. (Dr.Ing. Thesis)
UR-87-58	Magnus Småvik, MM	Thermal Load and Process Characteristics in a Two-Stroke Diesel Engine with Thermal Barriers (in Norwegian). (Dr.Ing. Thesis)
MTA-88-59	Bernt Arild Bremdal, MP	An Investigation of Marine Installation Processes – A Knowledge - Based Planning Approach. (Dr.Ing. Thesis)
MTA-88-60	Xu Jun, MK	Non-linear Dynamic Analysis of Space-framed Offshore Structures. (Dr.Ing. Thesis)
MTA-89-61	Gang Miao, MH	Hydrodynamic Forces and Dynamic Responses of Circular Cylinders in Wave Zones. (Dr.Ing. Thesis)
MTA-89-62	Martin Greenhow, MH	Linear and Non-Linear Studies of Waves and Floating Bodies. Part I and Part II. (Dr.Techn. Thesis)
MTA-89-63	Chang Li, MH	Force Coefficients of Spheres and Cubes in Oscillatory Flow with and without Current. (Dr.Ing. Thesis
MTA-89-64	Hu Ying, MP	A Study of Marketing and Design in Development of Marine Transport Systems. (Dr.Ing. Thesis)
MTA-89-65	Arild Jæger, MH	Seakeeping, Dynamic Stability and Performance of a Wedge Shaped Planing Hull. (Dr.Ing. Thesis)
MTA-89-66	Chan Siu Hung, MM	The dynamic characteristics of tilting-pad bearings
MTA-89-67	Kim Wikstrøm, MP	Analysis av projekteringen for ett offshore projekt. (Licenciat-avhandling)
MTA-89-68	Jiao Guoyang, MK	Reliability Analysis of Crack Growth under Random Loading, considering Model Updating. (Dr.Ing. Thesis)
MTA-89-69	Arnt Olufsen, MK	Uncertainty and Reliability Analysis of Fixed Offshore Structures. (Dr.Ing. Thesis)
MTA-89-70	Wu Yu-Lin, MR	System Reliability Analyses of Offshore Structures using improved Truss and Beam Models. (Dr.Ing. Thesis)
MTA-90-71	Jan Roger Hoff, MH	Three-dimensional Green function of a vessel with forward speed in waves. (Dr.Ing. Thesis)
MTA-90-72	Rong Zhao, MH	Slow-Drift Motions of a Moored Two-Dimensional Body in Irregular Waves. (Dr.Ing. Thesis)
MTA-90-73	Atle Minsaas, MP	Economical Risk Analysis. (Dr.Ing. Thesis)
MTA-90-74	Knut-Aril Farnes, MK	Long-term Statistics of Response in Non-linear Marine Structures. (Dr.Ing. Thesis)
MTA-90-75	Torbjørn Sotberg, MK	Application of Reliability Methods for Safety Assessment of Submarine Pipelines. (Dr.Ing.

		Thesis)
MTA-90-76	Zeuthen, Steffen, MP	SEAMAID. A computational model of the design process in a constraint-based logic programming environment. An example from the offshore domain. (Dr.Ing. Thesis)
MTA-91-77	Haagensen, Sven, MM	Fuel Dependant Cyclic Variability in a Spark Ignition Engine - An Optical Approach. (Dr.Ing. Thesis)
MTA-91-78	Løland, Geir, MH	Current forces on and flow through fish farms. (Dr.Ing. Thesis)
MTA-91-79	Hoen, Christopher, MK	System Identification of Structures Excited by Stochastic Load Processes. (Dr.Ing. Thesis)
MTA-91-80	Haugen, Stein, MK	Probabilistic Evaluation of Frequency of Collision between Ships and Offshore Platforms. (Dr.Ing. Thesis)
MTA-91-81	Sødahl, Nils, MK	Methods for Design and Analysis of Flexible Risers. (Dr.Ing. Thesis)
MTA-91-82	Ormberg, Harald, MK	Non-linear Response Analysis of Floating Fish Farm Systems. (Dr.Ing. Thesis)
MTA-91-83	Marley, Mark J., MK	Time Variant Reliability under Fatigue Degradation. (Dr.Ing. Thesis)
MTA-91-84	Krokstad, Jørgen R., MH	Second-order Loads in Multidirectional Seas. (Dr.Ing. Thesis)
MTA-91-85	Molteberg, Gunnar A., MM	The Application of System Identification Techniques to Performance Monitoring of Four Stroke Turbocharged Diesel Engines. (Dr.Ing. Thesis)
MTA-92-86	Mørch, Hans Jørgen Bjelke, MH	Aspects of Hydrofoil Design: with Emphasis on Hydrofoil Interaction in Calm Water. (Dr.Ing. Thesis)
MTA-92-87	Chan Siu Hung, MM	Nonlinear Analysis of Rotordynamic Instabilities in Highspeed Turbomachinery. (Dr.Ing. Thesis)
MTA-92-88	Bessason, Bjarni, MK	Assessment of Earthquake Loading and Response of Seismically Isolated Bridges. (Dr.Ing. Thesis)
MTA-92-89	Langli, Geir, MP	Improving Operational Safety through exploitation of Design Knowledge - an investigation of offshore platform safety. (Dr.Ing. Thesis)
MTA-92-90	Sævik, Svein, MK	On Stresses and Fatigue in Flexible Pipes. (Dr.Ing. Thesis)
MTA-92-91	Ask, Tor Ø., MM	Ignition and Flame Growth in Lean Gas-Air Mixtures. An Experimental Study with a Schlieren System. (Dr.Ing. Thesis)
MTA-86-92	Hessen, Gunnar, MK	Fracture Mechanics Analysis of Stiffened Tubular Members. (Dr.Ing. Thesis)

MTA-93-93	Steinebach, Christian, MM	Knowledge Based Systems for Diagnosis of Rotating Machinery. (Dr.Ing. Thesis)
MTA-93-94	Dalane, Jan Inge, MK	System Reliability in Design and Maintenance of Fixed Offshore Structures. (Dr.Ing. Thesis)
MTA-93-95	Steen, Sverre, MH	Cobblestone Effect on SES. (Dr.Ing. Thesis)
MTA-93-96	Karunakaran, Daniel, MK	Nonlinear Dynamic Response and Reliability Analysis of Drag-dominated Offshore Platforms. (Dr.Ing. Thesis)
MTA-93-97	Hagen, Arnulf, MP	The Framework of a Design Process Language. (Dr.Ing. Thesis)
MTA-93-98	Nordrik, Rune, MM	Investigation of Spark Ignition and Autoignition in Methane and Air Using Computational Fluid Dynamics and Chemical Reaction Kinetics. A Numerical Study of Ignition Processes in Internal Combustion Engines. (Dr.Ing. Thesis)
MTA-94-99	Passano, Elizabeth, MK	Efficient Analysis of Nonlinear Slender Marine Structures. (Dr.Ing. Thesis)
MTA-94-100	Kvålsvold, Jan, MH	Hydroelastic Modelling of Wetdeck Slamming on Multihull Vessels. (Dr.Ing. Thesis)
MTA-94-102	Bech, Sidsel M., MK	Experimental and Numerical Determination of Stiffness and Strength of GRP/PVC Sandwich Structures. (Dr.Ing. Thesis)
MTA-95-103	Paulsen, Hallvard, MM	A Study of Transient Jet and Spray using a Schlieren Method and Digital Image Processing. (Dr.Ing. Thesis)
MTA-95-104	Hovde, Geir Olav, MK	Fatigue and Overload Reliability of Offshore Structural Systems, Considering the Effect of Inspection and Repair. (Dr.Ing. Thesis)
MTA-95-105	Wang, Xiaozhi, MK	Reliability Analysis of Production Ships with Emphasis on Load Combination and Ultimate Strength. (Dr.Ing. Thesis)
MTA-95-106	Ulstein, Tore, MH	Nonlinear Effects of a Flexible Stern Seal Bag on Cobblestone Oscillations of an SES. (Dr.Ing. Thesis)
MTA-95-107	Solaas, Frøydis, MH	Analytical and Numerical Studies of Sloshing in Tanks. (Dr.Ing. Thesis)
MTA-95-108	Hellan, Øyvind, MK	Nonlinear Pushover and Cyclic Analyses in Ultimate Limit State Design and Reassessment of Tubular Steel Offshore Structures. (Dr.Ing. Thesis)
MTA-95-109	Hermundstad, Ole A., MK	Theoretical and Experimental Hydroelastic Analysis of High Speed Vessels. (Dr.Ing. Thesis)
MTA-96-110	Bratland, Anne K., MH	Wave-Current Interaction Effects on Large-Volume Bodies in Water of Finite Depth. (Dr.Ing. Thesis)
MTA-96-111	Herfjord, Kjell, MH	A Study of Two-dimensional Separated Flow by a Combination of the Finite Element Method and

		Navier-Stokes Equations. (Dr.Ing. Thesis)
MTA-96-112	Æsøy, Vilmar, MM	Hot Surface Assisted Compression Ignition in a Direct Injection Natural Gas Engine. (Dr.Ing. Thesis)
MTA-96-113	Eknes, Monika L., MK	Escalation Scenarios Initiated by Gas Explosions on Offshore Installations. (Dr.Ing. Thesis)
MTA-96-114	Erikstad, Stein O., MP	A Decision Support Model for Preliminary Ship Design. (Dr.Ing. Thesis)
MTA-96-115	Pedersen, Egil, MH	A Nautical Study of Towed Marine Seismic Streamer Cable Configurations. (Dr.Ing. Thesis)
MTA-97-116	Moksnes, Paul O., MM	Modelling Two-Phase Thermo-Fluid Systems Using Bond Graphs. (Dr.Ing. Thesis)
MTA-97-117	Halse, Karl H., MK	On Vortex Shedding and Prediction of Vortex-Induced Vibrations of Circular Cylinders. (Dr.Ing. Thesis)
MTA-97-118	Igland, Ragnar T., MK	Reliability Analysis of Pipelines during Laying, considering Ultimate Strength under Combined Loads. (Dr.Ing. Thesis)
MTA-97-119	Pedersen, Hans-P., MP	Levendefiskteknologi for fiskefartøy. (Dr.Ing. Thesis)
MTA-98-120	Vikestad, Kyrre, MK	Multi-Frequency Response of a Cylinder Subjected to Vortex Shedding and Support Motions. (Dr.Ing. Thesis)
MTA-98-121	Azadi, Mohammad R. E., MK	Analysis of Static and Dynamic Pile-Soil-Jacket Behaviour. (Dr.Ing. Thesis)
MTA-98-122	Ulltang, Terje, MP	A Communication Model for Product Information. (Dr.Ing. Thesis)
MTA-98-123	Torbergsen, Erik, MM	Impeller/Diffuser Interaction Forces in Centrifugal Pumps. (Dr.Ing. Thesis)
MTA-98-124	Hansen, Edmond, MH	A Discrete Element Model to Study Marginal Ice Zone Dynamics and the Behaviour of Vessels Moored in Broken Ice. (Dr.Ing. Thesis)
MTA-98-125	Videiro, Paulo M., MK	Reliability Based Design of Marine Structures. (Dr.Ing. Thesis)
MTA-99-126	Mainçon, Philippe, MK	Fatigue Reliability of Long Welds Application to Titanium Risers. (Dr.Ing. Thesis)
MTA-99-127	Haugen, Elin M., MH	Hydroelastic Analysis of Slamming on Stiffened Plates with Application to Catamaran Wetdecks. (Dr.Ing. Thesis)
MTA-99-128	Langhelle, Nina K., MK	Experimental Validation and Calibration of Nonlinear Finite Element Models for Use in Design of Aluminium Structures Exposed to Fire. (Dr.Ing. Thesis)
MTA-99-	Berstad, Are J., MK	Calculation of Fatigue Damage in Ship Structures.

129		(Dr.Ing. Thesis)
MTA-99-130	Andersen, Trond M., MM	Short Term Maintenance Planning. (Dr.Ing. Thesis)
MTA-99-131	Tveiten, Bård Wathne, MK	Fatigue Assessment of Welded Aluminium Ship Details. (Dr.Ing. Thesis)
MTA-99-132	Søreide, Fredrik, MP	Applications of underwater technology in deep water archaeology. Principles and practice. (Dr.Ing. Thesis)
MTA-99-133	Tønnessen, Rune, MH	A Finite Element Method Applied to Unsteady Viscous Flow Around 2D Blunt Bodies With Sharp Corners. (Dr.Ing. Thesis)
MTA-99-134	Elvekrok, Dag R., MP	Engineering Integration in Field Development Projects in the Norwegian Oil and Gas Industry. The Supplier Management of Norne. (Dr.Ing. Thesis)
MTA-99-135	Fagerholt, Kjetil, MP	Optimeringsbaserte Metoder for Ruteplanlegging innen skipsfart. (Dr.Ing. Thesis)
MTA-99-136	Bysveen, Marie, MM	Visualization in Two Directions on a Dynamic Combustion Rig for Studies of Fuel Quality. (Dr.Ing. Thesis)
MTA-2000-137	Storteig, Eskild, MM	Dynamic characteristics and leakage performance of liquid annular seals in centrifugal pumps. (Dr.Ing. Thesis)
MTA-2000-138	Sagli, Gro, MK	Model uncertainty and simplified estimates of long term extremes of hull girder loads in ships. (Dr.Ing. Thesis)
MTA-2000-139	Tronstad, Harald, MK	Nonlinear analysis and design of cable net structures like fishing gear based on the finite element method. (Dr.Ing. Thesis)
MTA-2000-140	Kroneberg, André, MP	Innovation in shipping by using scenarios. (Dr.Ing. Thesis)
MTA-2000-141	Haslum, Herbjørn Alf, MH	Simplified methods applied to nonlinear motion of spar platforms. (Dr.Ing. Thesis)
MTA-2001-142	Samdal, Ole Johan, MM	Modelling of Degradation Mechanisms and Stressor Interaction on Static Mechanical Equipment Residual Lifetime. (Dr.Ing. Thesis)
MTA-2001-143	Baarholm, Rolf Jarle, MH	Theoretical and experimental studies of wave impact underneath decks of offshore platforms. (Dr.Ing. Thesis)
MTA-2001-144	Wang, Lihua, MK	Probabilistic Analysis of Nonlinear Wave-induced Loads on Ships. (Dr.Ing. Thesis)
MTA-2001-145	Kristensen, Odd H. Holt, MK	Ultimate Capacity of Aluminium Plates under Multiple Loads, Considering HAZ Properties. (Dr.Ing. Thesis)
MTA-2001-146	Greco, Marilena, MH	A Two-Dimensional Study of Green-Water

		Loading. (Dr.Ing. Thesis)
MTA-2001-147	Heggelund, Svein E., MK	Calculation of Global Design Loads and Load Effects in Large High Speed Catamarans. (Dr.Ing. Thesis)
MTA-2001-148	Babalola, Olusegun T., MK	Fatigue Strength of Titanium Risers – Defect Sensitivity. (Dr.Ing. Thesis)
MTA-2001-149	Mohammed, Abuu K., MK	Nonlinear Shell Finite Elements for Ultimate Strength and Collapse Analysis of Ship Structures. (Dr.Ing. Thesis)
MTA-2002-150	Holmedal, Lars E., MH	Wave-current interactions in the vicinity of the sea bed. (Dr.Ing. Thesis)
MTA-2002-151	Rognebakke, Olav F., MH	Sloshing in rectangular tanks and interaction with ship motions. (Dr.Ing. Thesis)
MTA-2002-152	Lader, Pål Furset, MH	Geometry and Kinematics of Breaking Waves. (Dr.Ing. Thesis)
MTA-2002-153	Yang, Qinzheng, MH	Wash and wave resistance of ships in finite water depth. (Dr.Ing. Thesis)
MTA-2002-154	Melhus, Øyvin, MM	Utilization of VOC in Diesel Engines. Ignition and combustion of VOC released by crude oil tankers. (Dr.Ing. Thesis)
MTA-2002-155	Ronæss, Marit, MH	Wave Induced Motions of Two Ships Advancing on Parallel Course. (Dr.Ing. Thesis)
MTA-2002-156	Økland, Ole D., MK	Numerical and experimental investigation of whipping in twin hull vessels exposed to severe wet deck slamming. (Dr.Ing. Thesis)
MTA-2002-157	Ge, Chunhua, MK	Global Hydroelastic Response of Catamarans due to Wet Deck Slamming. (Dr.Ing. Thesis)
MTA-2002-158	Byklum, Eirik, MK	Nonlinear Shell Finite Elements for Ultimate Strength and Collapse Analysis of Ship Structures. (Dr.Ing. Thesis)
IMT-2003-1	Chen, Haibo, MK	Probabilistic Evaluation of FPSO-Tanker Collision in Tandem Offloading Operation. (Dr.Ing. Thesis)
IMT-2003-2	Skaugset, Kjetil Bjørn, MK	On the Suppression of Vortex Induced Vibrations of Circular Cylinders by Radial Water Jets. (Dr.Ing. Thesis)
IMT-2003-3	Chezhian, Muthu	Three-Dimensional Analysis of Slamming. (Dr.Ing. Thesis)
IMT-2003-4	Buhaug, Øyvind	Deposit Formation on Cylinder Liner Surfaces in Medium Speed Engines. (Dr.Ing. Thesis)
IMT-2003-5	Tregde, Vidar	Aspects of Ship Design: Optimization of Aft Hull with Inverse Geometry Design. (Dr.Ing. Thesis)
IMT-2003-6	Wist, Hanne Therese	Statistical Properties of Successive Ocean Wave Parameters. (Dr.Ing. Thesis)

IMT-2004-7	Ransau, Samuel	Numerical Methods for Flows with Evolving Interfaces. (Dr.Ing. Thesis)
IMT-2004-8	Soma, Torkel	Blue-Chip or Sub-Standard. A data interrogation approach of identity safety characteristics of shipping organization. (Dr.Ing. Thesis)
IMT-2004-9	Ersdal, Svein	An experimental study of hydrodynamic forces on cylinders and cables in near axial flow. (Dr.Ing. Thesis)
IMT-2005-10	Brodtkorb, Per Andreas	The Probability of Occurrence of Dangerous Wave Situations at Sea. (Dr.Ing. Thesis)
IMT-2005-11	Yttervik, Rune	Ocean current variability in relation to offshore engineering. (Dr.Ing. Thesis)
IMT-2005-12	Fredheim, Arne	Current Forces on Net-Structures. (Dr.Ing. Thesis)
IMT-2005-13	Heggernes, Kjetil	Flow around marine structures. (Dr.Ing. Thesis
IMT-2005-14	Fouques, Sebastien	Lagrangian Modelling of Ocean Surface Waves and Synthetic Aperture Radar Wave Measurements. (Dr.Ing. Thesis)
IMT-2006-15	Holm, Håvard	Numerical calculation of viscous free surface flow around marine structures. (Dr.Ing. Thesis)
IMT-2006-16	Bjørheim, Lars G.	Failure Assessment of Long Through Thickness Fatigue Cracks in Ship Hulls. (Dr.Ing. Thesis)
IMT-2006-17	Hansson, Lisbeth	Safety Management for Prevention of Occupational Accidents. (Dr.Ing. Thesis)
IMT-2006-18	Zhu, Xinying	Application of the CIP Method to Strongly Nonlinear Wave-Body Interaction Problems. (Dr.Ing. Thesis)
IMT-2006-19	Reite, Karl Johan	Modelling and Control of Trawl Systems. (Dr.Ing. Thesis)
IMT-2006-20	Smogeli, Øyvind Notland	Control of Marine Propellers. From Normal to Extreme Conditions. (Dr.Ing. Thesis)
IMT-2007-21	Storhaug, Gaute	Experimental Investigation of Wave Induced Vibrations and Their Effect on the Fatigue Loading of Ships. (Dr.Ing. Thesis)
IMT-2007-22	Sun, Hui	A Boundary Element Method Applied to Strongly Nonlinear Wave-Body Interaction Problems. (PhD Thesis, CeSOS)
IMT-2007-23	Rustad, Anne Marthine	Modelling and Control of Top Tensioned Risers. (PhD Thesis, CeSOS)
IMT-2007-24	Johansen, Vegar	Modelling flexible slender system for real-time simulations and control applications
IMT-2007-25	Wroldsen, Anders Sunde	Modelling and control of tensegrity structures. (PhD Thesis, CeSOS)
IMT-	Aronsen, Kristoffer Høye	An experimental investigation of in-line and

2007-26		combined inline and cross flow vortex induced vibrations. (Dr. avhandling, IMT)
IMT-2007-27	Gao, Zhen	Stochastic Response Analysis of Mooring Systems with Emphasis on Frequency-domain Analysis of Fatigue due to Wide-band Response Processes (PhD Thesis, CeSOS)
IMT-2007-28	Thorstensen, Tom Anders	Lifetime Profit Modelling of Ageing Systems Utilizing Information about Technical Condition. (Dr.ing. thesis, IMT)
IMT-2008-29	Refsnes, Jon Erling Gorset	Nonlinear Model-Based Control of Slender Body AUVs (PhD Thesis, IMT)
IMT-2008-30	Berntsen, Per Ivar B.	Structural Reliability Based Position Mooring. (PhD-Thesis, IMT)
IMT-2008-31	Ye, Naiquan	Fatigue Assessment of Aluminium Welded Box-stiffener Joints in Ships (Dr.ing. thesis, IMT)
IMT-2008-32	Radan, Damir	Integrated Control of Marine Electrical Power Systems. (PhD-Thesis, IMT)
IMT-2008-33	Thomassen, Paul	Methods for Dynamic Response Analysis and Fatigue Life Estimation of Floating Fish Cages. (Dr.ing. thesis, IMT)
IMT-2008-34	Pákozdi, Csaba	A Smoothed Particle Hydrodynamics Study of Two-dimensional Nonlinear Sloshing in Rectangular Tanks. (Dr.ing.thesis, IMT/ CeSOS)
IMT-2007-35	Grytøyr, Guttorm	A Higher-Order Boundary Element Method and Applications to Marine Hydrodynamics. (Dr.ing.thesis, IMT)
IMT-2008-36	Drummen, Ingo	Experimental and Numerical Investigation of Nonlinear Wave-Induced Load Effects in Containerships considering Hydroelasticity. (PhD thesis, CeSOS)
IMT-2008-37	Skejic, Renato	Maneuvering and Seakeeping of a Singel Ship and of Two Ships in Interaction. (PhD-Thesis, CeSOS)
IMT-2008-38	Harlem, Alf	An Age-Based Replacement Model for Repairable Systems with Attention to High-Speed Marine Diesel Engines. (PhD-Thesis, IMT)
IMT-2008-39	Alsos, Hagbart S.	Ship Grounding. Analysis of Ductile Fracture, Bottom Damage and Hull Girder Response. (PhD-thesis, IMT)
IMT-2008-40	Graczyk, Mateusz	Experimental Investigation of Sloshing Loading and Load Effects in Membrane LNG Tanks Subjected to Random Excitation. (PhD-thesis, CeSOS)
IMT-2008-41	Taghipour, Reza	Efficient Prediction of Dynamic Response for Flexible amd Multi-body Marine Structures. (PhD-thesis, CeSOS)
IMT-2008-42	Ruth, Eivind	Propulsion control and thrust allocation on marine

		vessels. (PhD thesis, CeSOS)
IMT-2008-43	Nystad, Bent Helge	Technical Condition Indexes and Remaining Useful Life of Aggregated Systems. PhD thesis, IMT
IMT-2008-44	Soni, Prashant Kumar	Hydrodynamic Coefficients for Vortex Induced Vibrations of Flexible Beams, PhD thesis, CeSOS
IMT-2009-45	Amlashi, Hadi K.K.	Ultimate Strength and Reliability-based Design of Ship Hulls with Emphasis on Combined Global and Local Loads. PhD Thesis, IMT
IMT-2009-46	Pedersen, Tom Arne	Bond Graph Modelling of Marine Power Systems. PhD Thesis, IMT
IMT-2009-47	Kristiansen, Trygve	Two-Dimensional Numerical and Experimental Studies of Piston-Mode Resonance. PhD-Thesis, CeSOS
IMT-2009-48	Ong, Muk Chen	Applications of a Standard High Reynolds Number Model and a Stochastic Scour Prediction Model for Marine Structures. PhD-thesis, IMT
IMT-2009-49	Hong, Lin	Simplified Analysis and Design of Ships subjected to Collision and Grounding. PhD-thesis, IMT
IMT-2009-50	Koushan, Kamran	Vortex Induced Vibrations of Free Span Pipelines, PhD thesis, IMT
IMT-2009-51	Korsvik, Jarl Eirik	Heuristic Methods for Ship Routing and Scheduling. PhD-thesis, IMT
IMT-2009-52	Lee, Jihoon	Experimental Investigation and Numerical in Analyzing the Ocean Current Displacement of Longlines. Ph.d.-Thesis, IMT.
IMT-2009-53	Vestbøstad, Tone Gran	A Numerical Study of Wave-in-Deck Impact usin a Two-Dimensional Constrained Interpolation Profile Method, Ph.d.thesis, CeSOS.
IMT-2009-54	Bruun, Kristine	Bond Graph Modelling of Fuel Cells for Marine Power Plants. Ph.d.-thesis, IMT
IMT 2009-55	Holstad, Anders	Numerical Investigation of Turbulence in a Sekwed Three-Dimensional Channel Flow, Ph.d.-thesis, IMT.
IMT 2009-56	Ayala-Uraga, Efren	Reliability-Based Assessment of Deteriorating Ship-shaped Offshore Structures, Ph.d.-thesis, IMT
IMT 2009-57	Kong, Xiangjun	A Numerical Study of a Damaged Ship in Beam Sea Waves. Ph.d.-thesis, IMT/CeSOS.
IMT 2010-58	Kristiansen, David	Wave Induced Effects on Floaters of Aquaculture Plants, Ph.d.-thesis, CeSOS.
IMT 2010-59	Ludvigsen, Martin	An ROV-Toolbox for Optical and Acoustic Scientific Seabed Investigation. Ph.d.-thesis IMT.

IMT 2010-60	Hals, Jørgen	Modelling and Phase Control of Wave-Energy Converters. Ph.d.thesis, CeSOS.
IMT 2010- 61	Shu, Zhi	Uncertainty Assessment of Wave Loads and Ultimate Strength of Tankers and Bulk Carriers in a Reliability Framework. Ph.d. Thesis, IMT/ CeSOS
IMT 2010-62	Shao, Yanlin	Numerical Potential-Flow Studies on Weakly-Nonlinear Wave-Body Interactions with/without Small Forward Speed, Ph.d.thesis,CeSOS.
IMT 2010-63	Califano, Andrea	Dynamic Loads on Marine Propellers due to Intermittent Ventilation. Ph.d.thesis, IMT.
IMT 2010-64	El Khoury, George	Numerical Simulations of Massively Separated Turbulent Flows, Ph.d.-thesis, IMT
IMT 2010-65	Seim, Knut Sponheim	Mixing Process in Dense Overflows with Emphasis on the Faroe Bank Channel Overflow. Ph.d.thesis, IMT
IMT 2010-66	Jia, Huirong	Structural Analysis of Intect and Damaged Ships in a Collission Risk Analysis Perspective. Ph.d.thesis CeSoS.
IMT 2010-67	Jiao, Linlin	Wave-Induced Effects on a Pontoon-type Very Large Floating Structures (VLFS). Ph.D.-thesis, CeSOS.
IMT 2010-68	Abrahamsen, Bjørn Christian	Sloshing Induced Tank Roof with Entrapped Air Pocket. Ph.d.thesis, CeSOS.
IMT 2011-69	Karimirad, Madjid	Stochastic Dynamic Response Analysis of Spar-Type Wind Turbines with Catenary or Taut Mooring Systems. Ph.d.-thesis, CeSOS.
IMT - 2011-70	Erlend Meland	Condition Monitoring of Safety Critical Valves. Ph.d.-thesis, IMT.
IMT – 2011-71	Yang, Limin	Stochastic Dynamic System Analysis of Wave Energy Converter with Hydraulic Power Take-Off, with Particular Reference to Wear Damage Analysis, Ph.d. Thesis, CeSOS.
IMT – 2011-72	Visscher, Jan	Application of Particla Image Velocimetry on Turbulent Marine Flows, Ph.d.Thesis, IMT.
IMT – 2011-73	Su, Biao	Numerical Predictions of Global and Local Ice Loads on Ships. Ph.d.Thesis, CeSOS.
IMT – 2011-74	Liu, Zhenhui	Analytical and Numerical Analysis of Iceberg Collision with Ship Structures. Ph.d.Thesis, IMT.
IMT – 2011-75	Aarsæther, Karl Gunnar	Modeling and Analysis of Ship Traffic by Observation and Numerical Simulation. Ph.d.Thesis, IMT.
Imt – 2011-76	Wu, Jie	Hydrodynamic Force Identification from Stochastic Vortex Induced Vibration Experiments with Slender Beams. Ph.d.Thesis, IMT.

IMT – 2011-78	Nguyen, Tan-Hoi	Toward a System of Real-Time Prediction and Monitoring of Bottom Damage Conditions During Ship Grounding. Ph.d.thesis, IMT.
IMT-2011-79	Tavakoli, Mohammad T.	Assessment of Oil Spill in Ship Collision and Grounding, Ph.d.thesis, IMT.
IMT-2011-80	Guo, Bingjie	Numerical and Experimental Investigation of Added Resistance in Waves. Ph.d.Thesis, IMT.
IMT-2011-81	Chen, Qiaofeng	Ultimate Strength of Aluminium Panels, considering HAZ Effects, IMT
IMT-2012-82	Kota, Ravikiran S.	Wave Loads on Decks of Offshore Structures in Random Seas, CeSOS.
IMT-2012-83	Sten, Ronny	Dynamic Simulation of Deep Water Drilling Risers with Heave Compensating System, IMT.
IMT-2012-84	Berle, Øyvind	Risk and resilience in global maritime supply chains, IMT.
IMT-2012-85	Fang, Shaoji	Fault Tolerant Position Mooring Control Based on Structural Reliability, CeSOS.
IMT-2012-86	You, Jikun	Numerical studies on wave forces and moored ship motions in intermediate and shallow water, CeSOS.
IMT-2012-87	Xiang ,Xu	Maneuvering of two interacting ships in waves, CeSOS
IMT-2012-88	Dong, Wenbin	Time-domain fatigue response and reliability analysis of offshore wind turbines with emphasis on welded tubular joints and gear components, CeSOS
IMT-2012-89	Zhu, Suji	Investigation of Wave-Induced Nonlinear Load Effects in Open Ships considering Hull Girder Vibrations in Bending and Torsion, CeSOS
IMT-2012-90	Zhou, Li	Numerical and Experimental Investigation of Station-keeping in Level Ice, CeSOS
IMT-2012-91	Ushakov, Sergey	Particulate matter emission characteristics from diesel enignes operating on conventional and alternative marine fuels, IMT
IMT-2013-1	Yin, Decao	Experimental and Numerical Analysis of Combined In-line and Cross-flow Vortex Induced Vibrations, CeSOS
IMT-2013-2	Kurniawan, Adi	Modelling and geometry optimisation of wave energy converters, CeSOS
IMT-2013-3	Al Ryati, Nabil	Technical condition indexes doe auxiliary marine diesel engines, IMT
IMT-2013-4	Firoozkoohi, Reza	Experimental, numerical and analytical investigation of the effect of screens on sloshing, CeSOS

IMT-2013-5	Ommani, Babak	Potential-Flow Predictions of a Semi-Displacement Vessel Including Applications to Calm Water Broaching, CeSOS
IMT-2013-6	Xing, Yihan	Modelling and analysis of the gearbox in a floating spar-type wind turbine, CeSOS
IMT-7-2013	Balland, Océane	Optimization models for reducing air emissions from ships, IMT
IMT-8-2013	Yang, Dan	Transitional wake flow behind an inclined flat plate-----Computation and analysis, IMT
IMT-9-2013	Abdillah, Suyuthi	Prediction of Extreme Loads and Fatigue Damage for a Ship Hull due to Ice Action, IMT
IMT-10-2013	Ramìrez, Pedro Agustìn Pèrez	Ageing management and life extension of technical systems: Concepts and methods applied to oil and gas facilities, IMT
IMT-11-2013	Chuang, Zhenju	Experimental and Numerical Investigation of Speed Loss due to Seakeeping and Maneuvering. IMT
IMT-12-2013	Etemaddar, Mahmoud	Load and Response Analysis of Wind Turbines under Atmospheric Icing and Controller System Faults with Emphasis on Spar Type Floating Wind Turbines, IMT
IMT-13-2013	Lindstad, Haakon	Strategies and measures for reducing maritime CO2 emissons, IMT
IMT-14-2013	Haris, Sabril	Damage interaction analysis of ship collisions, IMT
IMT-15-2013	Shainee, Mohamed	Conceptual Design, Numerical and Experimental Investigation of a SPM Cage Concept for Offshore Mariculture, IMT
IMT-16-2013	Gansel, Lars	Flow past porous cylinders and effects of biofouling and fish behavior on the flow in and around Atlantic salmon net cages, IMT
IMT-17-2013	Gaspar, Henrique	Handling Aspects of Complexity in Conceptual Ship Design, IMT
IMT-18-2013	Thys, Maxime	Theoretical and Experimental Investigation of a Free Running Fishing Vessel at Small Frequency of Encounter, CeSOS
IMT-19-2013	Aglen, Ida	VIV in Free Spanning Pipelines, CeSOS
IMT-1-2014	Song, An	Theoretical and experimental studies of wave diffraction and radiation loads on a horizontally submerged perforated plate, CeSOS
IMT-2-2014	Rogne, Øyvind Ygre	Numerical and Experimental Investigation of a Hinged 5-body Wave Energy Converter, CeSOS
IMT-3-2014	Dai, Lijuan	Safe and efficient operation and maintenance of offshore wind farms ,IMT

IMT-4-2014	Bachynski, Erin Elizabeth	Design and Dynamic Analysis of Tension Leg Platform Wind Turbines, CeSOS
IMT-5-2014	Wang, Jingbo	Water Entry of Freefall Wedged – Wedge motions and Cavity Dynamics, CeSOS
IMT-6-2014	Kim, Ekaterina	Experimental and numerical studies related to the coupled behavior of ice mass and steel structures during accidental collisions, IMT
IMT-7-2014	Tan, Xiang	Numerical investigation of ship’s continuous- mode icebreaking in leverl ice, CeSOS
IMT-8-2014	Muliawan, Made Jaya	Design and Analysis of Combined Floating Wave and Wind Power Facilities, with Emphasis on Extreme Load Effects of the Mooring System, CeSOS
IMT-9-2014	Jiang, Zhiyu	Long-term response analysis of wind turbines with an emphasis on fault and shutdown conditions, IMT
IMT-10-2014	Dukan, Fredrik	ROV Motion Control Systems, IMT
IMT-11-2014	Grimsmo, Nils I.	Dynamic simulations of hydraulic cylinder for heave compensation of deep water drilling risers, IMT
IMT-12-2014	Kvittem, Marit I.	Modelling and response analysis for fatigue design of a semisubmersible wind turbine, CeSOS
IMT-13-2014	Akhtar, Juned	The Effects of Human Fatigue on Risk at Sea, IMT
IMT-14-2014	Syahroni, Nur	Fatigue Assessment of Welded Joints Taking into Account Effects of Residual Stress, IMT
IMT-1-2015	Bøckmann, Eirik	Wave Propulsion of ships, IMT
IMT-2-2015	Wang, Kai	Modelling and dynamic analysis of a semi-submersible floating vertical axis wind turbine, CeSOS
IMT-3-2015	Fredriksen, Arnt Gunvald	A numerical and experimental study of a two-dimensional body with moonpool in waves and current, CeSOS
IMT-4-2015	Jose Patricio Gallardo Canabes	Numerical studies of viscous flow around bluff bodies, IMT
IMT-5-2015	Vegard Longva	Formulation and application of finite element techniques for slender marine structures subjected to contact interactions, IMT
IMT-6-2015	Jacobus De Vaal	Aerodynamic modelling of floating wind turbines, CeSOS
IMT-7-2015	Fachri Nasution	Fatigue Performance of Copper Power Conductors, IMT

IMT-8-2015	Oleh I Karpa	Development of bivariate extreme value distributions for applications in marine technology,CeSOS
IMT-9-2015	Daniel de Almeida Fernandes	An output feedback motion control system for ROVs, AMOS
IMT-10-2015	Bo Zhao	Particle Filter for Fault Diagnosis: Application to Dynamic Positioning Vessel and Underwater Robotics, CeSOS
IMT-11-2015	Wenting Zhu	Impact of emission allocation in maritime transportation, IMT
IMT-12-2015	Amir Rasekhi Nejad	Dynamic Analysis and Design of Gearboxes in Offshore Wind Turbines in a Structural Reliability Perspective, CeSOS
IMT-13-2015	Arturo Jesùs Ortega Malca	Dynamic Response of Flexibles Risers due to Unsteady Slug Flow, CeSOS
IMT-14-2015	Dagfinn Husjord	Guidance and decision-support system for safe navigation of ships operating in close proximity, IMT
IMT-15-2015	Anirban Bhattacharyya	Ducted Propellers: Behaviour in Waves and Scale Effects, IMT
IMT-16-2015	Qin Zhang	Image Processing for Ice Parameter Identification in Ice Management, IMT
IMT-1-2016	Vincentius Rumawas	Human Factors in Ship Design and Operation: An Experiential Learning, IMT
IMT-2-2016	Martin Storheim	Structural response in ship-platform and ship-ice collisions, IMT
IMT-3-2016	Mia Abrahamsen Prsic	Numerical Simulations of the Flow around single and Tandem Circular Cylinders Close to a Plane Wall, IMT
IMT-4-2016	Tufan Arslan	Large-eddy simulations of cross-flow around ship sections, IMT
IMT-5-2016	Pierre Yves-Henry	Parametrisation of aquatic vegetation in hydraulic and coastal research,IMT
IMT-6-2016	Lin Li	Dynamic Analysis of the Instalation of Monopiles for Offshore Wind Turbines, CeSOS
IMT-7-2016	Øivind Kåre Kjerstad	Dynamic Positioning of Marine Vessels in Ice, IMT
IMT-8-2016	Xiaopeng Wu	Numerical Analysis of Anchor Handling and Fish Trawling Operations in a Safety Perspective, CeSOS

IMT-9-2016	Zhengshun Cheng	Integrated Dynamic Analysis of Floating Vertical Axis Wind Turbines, CeSOS
IMT-10-2016	Ling Wan	Experimental and Numerical Study of a Combined Offshore Wind and Wave Energy Converter Concept
IMT-11-2016	Wei Chai	Stochastic dynamic analysis and reliability evaluation of the roll motion for ships in random seas, CeSOS
IMT-12-2016	Øyvind Selnes Patricksson	Decision support for conceptual ship design with focus on a changing life cycle and future uncertainty, IMT
IMT-13-2016	Mats Jørgen Thorsen	Time domain analysis of vortex-induced vibrations, IMT
IMT-14-2016	Edgar McGuinness	Safety in the Norwegian Fishing Fleet – Analysis and measures for improvement, IMT
IMT-15-2016	Sepideh Jafarzadeh	Energy effiency and emission abatement in the fishing fleet, IMT
IMT-16-2016	Wilson Ivan Guachamin Acero	Assessment of marine operations for offshore wind turbine installation with emphasis on response-based operational limits, IMT
IMT-17-2016	Mauro Candeloro	Tools and Methods for Autonomous Operations on Seabed and Water Coumn using Underwater Vehicles, IMT
IMT-18-2016	Valentin Chabaud	Real-Time Hybrid Model Testing of Floating Wind Tubines, IMT
IMT-1-2017	Mohammad Saud Afzal	Three-dimensional streaming in a sea bed boundary layer
IMT-2-2017	Peng Li	A Theoretical and Experimental Study of Wave-induced Hydroelastic Response of a Circular Floating Collar
IMT-3-2017	Martin Bergström	A simulation-based design method for arctic maritime transport systems
IMT-4-2017	Bhushan Taskar	The effect of waves on marine propellers and propulsion
IMT-5-2017	Mohsen Bardestani	A two-dimensional numerical and experimental study of a floater with net and sinker tube in waves and current
IMT-6-2017	Fatemeh Hoseini Dadmarzi	Direct Numerical Simualtion of turbulent wakes behind different plate configurations
IMT-7-2017	Michel R. Miyazaki	Modeling and control of hybrid marine power plants

IMT-8-2017	Giri Rajasekhar Gunnu	Safety and effiency enhancement of anchor handling operations with particular emphasis on the stability of anchor handling vessels
IMT-9-2017	Kevin Koosup Yum	Transient Performance and Emissions of a Turbocharged Diesel Engine for Marine Power Plants
IMT-10-2017	Zhaolong Yu	Hydrodynamic and structural aspects of ship collisions
IMT-11-2017	Martin Hassel	Risk Analysis and Modelling of Allisions between Passing Vessels and Offshore Installations
IMT-12-2017	Astrid H. Brodtkorb	Hybrid Control of Marine Vessels – Dynamic Positioning in Varying Conditions
IMT-13-2017	Kjersti Bruserud	Simultaneous stochastic model of waves and current for prediction of structural design loads
IMT-14-2017	Finn-Idar Grøtta Giske	Long-Term Extreme Response Analysis of Marine Structures Using Inverse Reliability Methods
IMT-15-2017	Stian Skjong	Modeling and Simulation of Maritime Systems and Operations for Virtual Prototyping using co-Simulations