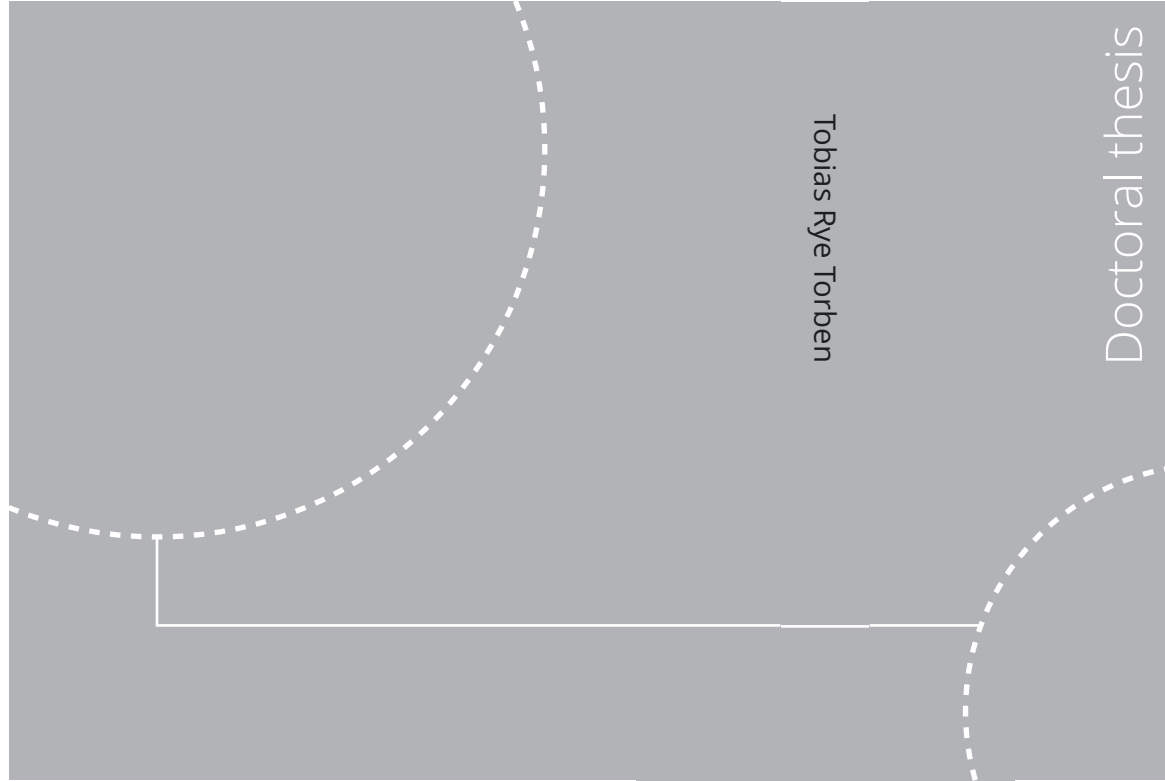


ISBN 978-82-326-5586-1 (printed ver.)
ISBN 978-82-326-6902-8 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)



Doctoral theses at NTNU, 2023:70

Tobias Rye Torben

Formal approaches to design
and verification of safe control
systems for autonomous
vessels

Doctoral theses at NTNU, 2023:70

NTNU
Norwegian University of
Science and Technology
Thesis for the degree of
Philosophiae Doctor
Faculty of Engineering
Department of Marine Technology

 **NTNU**
Norwegian University of
Science and Technology

 NTNU

 **NTNU**
Norwegian University of
Science and Technology

Tobias Rye Torben

Formal approaches to design and verification of safe control systems for autonomous vessels

Thesis for the degree of Philosophiae Doctor

Trondheim, March 2023

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



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 Marine Technology

© Tobias Rye Torben

ISBN 978-82-326-5586-1 (printed ver.)
ISBN 978-82-326-6902-8 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

Doctoral theses at NTNU, 2023:70



Printed by Skipnes Kommunikasjon AS

Summary

Maritime autonomy can positively impact society by cutting costs and emissions while enabling new solutions for transportation and mobility. Autonomous vessels must be capable of performing complex tasks under significant uncertainty in an unstructured environment, which requires fundamental innovations in the control systems. Introducing fundamentally new technology, combined with increased system complexity and criticality, introduces new risks that must be identified and mitigated to ensure safety. The success of maritime autonomy is ultimately hinged upon whether autonomy developers, regulators, and classification societies can find tractable solutions for safety assurance of autonomous vessel control systems.

Formal Methods (FMs) are a family of mathematically based methods for design and verification that have been used actively for assurance of safety-critical systems in several other industries. Recently, there has also been active research on FMs applied to autonomous systems. The maritime industry has, however, no tradition of using FMs. This thesis investigates how formal approaches, such as FMs, can contribute to solving the safety assurance challenges for autonomous vessel control systems.

The thesis first aims to identify the key challenges to be solved. A root cause of the identified safety assurance challenges is that autonomous systems extensively sense and interact with the open environment. This characteristic effectively means that an autonomous system may encounter infinitely many unknown scenarios that are impossible to specify completely during design. This is challenging in itself but has also led to the use of machine learning (ML) algorithms that learn from data instead of being programmed from a specification, which in turn introduces several challenges for safety assurance. Autonomous vessel control systems also have high internal complexity in addition to being software-intensive and safety-critical. Understanding all the interactions, failure mechanisms and system-level emergent behaviors is almost impossible to do *á priori*. Moreover, this internal complexity further adds to the huge span of possible scenarios. Some maritime-specific challenges are also identified, such as uncertainties in the motion control, the presence of complex power and propulsion systems that require significant monitoring and control, and the limited access to operational experience compared to other industries.

Two key needs for addressing the challenges are identified. Firstly, increased formality is needed in order to manage and limit the complex interactions and obtain

sufficient system integrity for the safety-critical application. Secondly, extensive use of simulation-based testing is necessary to be able to assess the system-level behavior in a scalable manner, such that the huge scenario space is sufficiently covered.

The research papers of the thesis propose novel methodologies to address these needs. Some key contributions are:

- A methodology which uses the formal logic STL in conjunction with a Gaussian process model to formalize and automate simulation-based testing. This enables simulation-based testing at a massive scale, with quantification of the test space coverage and confidence level.
- A methodology for contract-based design and verification which combines simulation-based testing with formal compositional reasoning using an automated theorem prover.
- The use of hybrid systems theory for formulation and formal stability analysis of a novel resetting observer design.

In addition to developing new methodology, the thesis reviews the literature on FMs and identifies several existing tools and methodologies which have the potential to address the safety assurance challenges for autonomous vessel control systems. Some key identified opportunities are formal specification, code generation, correct-by-construction controller synthesis, model checking of supervisory control logic, and automated theorem proving of motion planning algorithms.

An additional research objective of this thesis is to improve the control system performance of autonomous vessels by improving the guidance, navigation, and control (GNC) systems. GNC functionality forms the foundation on which autonomy functionality is built, and it is therefore vital for safe and robust autonomy. Moreover, uncertainties in the motion control were identified as a key challenge for safety assurance of maritime autonomy, and improvement of GNC functionality is key to reducing this uncertainty. The thesis contributes to this research objective by developing a more reactive observer design, and by developing a novel control allocation algorithm for double-ended ferries, which has been important in the development of the autonomous ferry pilot milliAmpere.

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of philosophiae doctor (PhD) at the Norwegian University of Science and Technology (NTNU). The work has been conducted at the Department of Marine Technology (IMT), under the Centre for Autonomous Marine Operations and Systems (AMOS) (RCN project number 223254). My main supervisor has been Professor Asgeir J. Sørensen from IMT, and my co-supervisors have been Professor Ingrid B. Utne from IMT and Professor Tor Arne Johansen from the Department of Engineering Cybernetics, NTNU.

The work has been part of the KPN project Online Risk Management and Risk Control for Autonomous Ships (ORCAS) (RCN project number 280655). My work is one of three PhDs in this project. My colleague Thomas Johansen's PhD has focused on online risk modeling, and my colleague Simon Blindheim's PhD has focused on risk-based model predictive control (MPC). The ORCAS project also has the classification society DNV and the technology provider Kongsberg Maritime as industry partners.

I started as an integrated PhD candidate during the final year of my master's studies in 2018. My work started with a practical summer project, where I worked on developing a motion control system for NTNU's autonomous ferry prototype, milliAmpere. This work resulted in my first publication, a paper on control allocation for double-ended ferries. After delivering my master's thesis, I started my 3-year PhD program in 2019. During my first year, I was situated in Trondheim. For my second year, I had a Fulbright Scholarship to be a visiting researcher in Professor Murat Arcaç's research group at the University of California Berkeley. Sadly, this was not possible to carry out due to the covid pandemic. Instead, I moved to Oslo, where I spent the final two years of my PhD, partly working from home due to the pandemic, and partly working from DNV's offices in Høvik.

Acknowledgments

I would like to give special thanks to my main supervisor, Asgeir for being a great mentor, motivator, and source of inspiration. Despite a busy schedule, Asgeir has found the time to follow me closely throughout my entire PhD period and has provided good advice for work and life in general. I would also like to thank my co-supervisors. Ingrid has also followed me closely and provided valuable input and guidance, Tor Arne has shared his expertise and provided good input and

discussions at our ORCAS project meetings. A great thanks also goes out to all my fellow PhD candidates at IMT and AMOS.

I would also like to thank the ORCAS industry partners from Kongsberg and DNV. In particular, Jon Arne Glomsrud (DNV) and Tom Arne Pedersen (DNV), for many good discussions which resulted in several joint publications. I am also grateful to the people at DNV's maritime research group for hosting me at their offices for the final two years of my PhD.

Thanks to my parents, Sverre and Katrin, for always supporting me, encouraging me, showing an interest in my work, and making me feel safe in my choices. Finally, a special thanks goes to my fiancé, Katrine, and my daughter, Sophia for their continuous love, support, and patience during many late nights and weekends when writing this thesis.

Oslo, November 28th 2022

Tobias Rye Torben

Contents

Summary	i
Preface	iii
Contents	v
List of Abbreviations	vii
List of Figures	ix
I Thesis Overview and Background	1
1 Introduction	3
1.1 Motivation and Background	3
1.2 Research Questions and Objectives	6
1.3 Research Methodology	6
1.4 Contributions at a Glance	6
1.5 Thesis Organization and Overview	10
2 Autonomous Vessels	13
2.1 What is Autonomy?	13
2.2 Control System Design	15
3 Safety Assurance	19
3.1 Safety Assurance of Autonomy	19
3.2 Hazard Analysis and Risk Assessment	20
4 Verification	23
4.1 Verification of Cyber-physical Systems	23
4.2 Simulation-based Verification	24
4.3 Formal Verification	26
4.4 Analytical Verification	26
5 Formal Methods	27
5.1 Introduction and Overview	27
5.2 Preliminaries	29

5.3	Temporal Logic	30
5.4	Contract-based Design	33
5.5	Automated Theorem Proving	35
6	Hybrid Systems	37
6.1	Hybrid Dynamical Systems Framework	37
6.2	Stability and Robustness	39
7	Discussion of Results	41
7.1	Research Question 1	41
7.2	Research Question 2	44
7.3	Research Question 3	46
8	Conclusions and Future Work	49
8.1	Conclusions	49
8.2	Future Work	51
	References	53
II	Selected Publications	61
	Paper A: Automatic simulation-based testing of autonomous ships using Gaussian processes and temporal logic	63
	Paper B: Towards Contract-based Verification for Autonomous Vessels	87
	Paper C: Resetting observer design for linear time-varying systems with application to dynamic positioning of marine surface vessels	115
	Paper D: On Formal Methods for Design and Verification of Maritime Autonomous Surface Ships	125
	Paper E: Evolution Of Safety In Marine Systems: From STPA To Auto- mated Test Scenario Generation	137
	Paper F: Development and testing of a risk-based control system for au- tonomous ships	155
	Paper G: Control-allocation for Double-ended Ferries with Full-scale Ex- perimental Results	185
	Previous PhD Theses Published at the Department of Marine Technology	195

List of Abbreviations

A/D Analog-to-Digital.

BBN Bayesian Belief Network.

CAPEX Capital Expenditures.

COLREG Convention on the International Regulations for Preventing Collisions at Sea.

D/A Digital-to-Analog.

DAE Differential Algebraic Equation.

EMS Energy Management System.

ENC Electronic Navigational Chart.

ETA Event Tree Analysis.

FMEA Failure Mode and Effect Analysis.

FMs Formal Methods.

FTA Fault Tree Analysis.

GNC Guidance, Navigation and Control.

GNSS Global Navigation Satellite System.

GP Gaussian Process.

GpAS Global pre-asymptotic Stability.

HARA Hazard Analysis and Risk Assessment.

HAZID Hazard Identification.

HiL Hardware-in-the-Loop.

IMO International Maritime Organization.

IMU Inertial Measurement Unit.

IR Infrared.

LB Locally bounded.

LTL Linear Temporal Logic.

LTV Linear Time-varying.

MiL Model-in-the-Loop.

ML Machine Learning.

MRC Minimum Risk Condition.

MTL Metric Temporal Logic.

NP Nondeterministic Polynomial.

OPEX Operating Expenses.

OSC Outer semicontinuous.

PHA Preliminary Hazard Analysis.

PMS Power Management System.

SiL Software-in-the-Loop.

SMT Satisfiability Modulo Theories.

SOTIF Safety of the Intended Functionality.

STL Signal Temporal Logic.

STPA Systems Theoretic Process Analysis.

UCA Unsafe Control Action.

UGpAS Uniform Global pre-asymptotic Stability.

List of Figures

1.1	ASKO and Kongsberg Maritime have constructed two small, autonomous, and zero-emission cargo vessels capable of carrying 16 trailers across Oslofjorden in Norway. Their operation will remove 150 daily diesel truck trips from the congested roads in the Oslo area. Photo: Courtesy of Kongsberg Maritime.	4
1.2	The autonomy start-up Zeabuz develops flexible networks of small autonomous ferries for urban waterborne mobility. This is an artistic illustration from an operation in Singapore. Photo: Courtesy of Zeabuz.	4
2.1	The nested feedback loops in a conventional vessel. The outermost loop is manual and performed by the onboard crew.	14
2.2	The nested feedback loops in an autonomous vessel. The outer loop is automated by an autonomy system, enabling the human operator to be removed from the real-time control loop.	15
2.3	Functional architecture for autonomous vessel control systems.	16
3.1	The four steps of an STPA. Figure redrawn based on similar figure from Leveson and Thomas (2018).	21
4.1	A spectrum of verification methods for cyber-physical systems. The horizontal axis indicates the level of exhaustiveness. The vertical axis indicates the scalability of the method. Source: Paper A	24
4.2	Spectrum of the models used by the verification methods from Figure 4.1. The models are ranked according to their fidelity, that is, how closely their behavior resembles the behavior of the real implementation that they model. Source: Paper A	25
5.1	Steps and activities in a formal development process. From Paper D	28

Part I

Thesis Overview and Background

Chapter 1

Introduction

1.1 Motivation and Background

The past decade has seen a strong push towards autonomous transportation solutions across several industries, including the maritime. Autonomy may reduce operating expenses (OPEX) for traditional ships by crew reductions and has the potential to reduce both OPEX and greenhouse gas emissions by enabling energy optimizations, such as slow steaming, optimal route planning, and optimal control. Autonomy may also reduce capital expenditures (CAPEX) by removing the need for infrastructure for human accommodations (DNV, 2018). There is also an acute global shortage of trained seafarers (CBC, 2022; NRK, 2022), and autonomy may be a key solution to address this.

In addition to optimizing traditional ships, autonomy may also enable completely new modes of transportation. For instance, autonomy can enable flexible networks of small unmanned vessels which are supervised by a single remote operator. Such solutions can make it economically feasible and operationally practical to move the transportation of people and goods away from congested roads and onto the underutilized waterways. Figures 1.1 and 1.2 show two examples of new modes of transportation enabled by autonomy.

Another claimed benefit of autonomous vessels is increased safety. The rationale behind this claim is typically that human operators are the cause of the majority of accidents, and reducing the dependency on human operators therefore will reduce the number of accidents (Porathe et al., 2018). While it seems reasonable that autonomy will reduce or remove some risks, it will certainly also introduce new potential risks that must be identified and mitigated to achieve the sought-after safety gains. The success of maritime autonomy is ultimately hinged upon whether autonomy developers, regulators, and classification societies can find tractable solutions for safety assurance.

There are many aspects to consider for the safety assurance of autonomous vessels, but fundamentally their novelty is related to the enhanced capabilities of



Figure 1.1: ASKO and Kongsberg Maritime have constructed two small, autonomous, and zero-emission cargo vessels capable of carrying 16 trailers across Oslofjorden in Norway. Their operation will remove 150 daily diesel truck trips from the congested roads in the Oslo area. Photo: Courtesy of Kongsberg Maritime.



Figure 1.2: The autonomy start-up Zeabuz develops flexible networks of small autonomous ferries for urban waterborne mobility. This is an artistic illustration from an operation in Singapore. Photo: Courtesy of Zeabuz.

their control systems. At their base, autonomous vessel control systems include advanced automation systems for guidance, navigation, and control (GNC) as well as monitoring and control of power and propulsion systems. Building on top of this, autonomous vessels must additionally be capable of obtaining situational awareness and performing intelligent planning and decision-making under uncertainty in the dynamic and unstructured maritime environment. The realization of au-

tonomous vessel control systems is enabled by recent advances in optimization, planning, artificial intelligence, computer vision, and sensor fusion, in addition to the ever-increasing computational resources available for embedded systems. While these technological advances enable the necessary functionality for autonomy, combining all of them into an integrated system, layered on top of the already complex maritime control systems, results in immense system complexity. Safety assurance of complex cyber-physical systems is a well-known challenge (Haugen, 2019; Sangiovanni-Vincentelli et al., 2012). In addition to the sheer system complexity, autonomy also comes with its own set of unique challenges for safety assurance due to the extensive sensing and interaction with the open environment. This effectively means that there will be infinitely many unknown scenarios that an autonomous vessel may encounter during planned operation, and it will thus never be possible to specify the required behavior in every scenario during design.

There is a clear need for a new methodology for the design and verification of safe control systems for autonomous vessels. A possible approach to address the safety assurance challenges is to increase the formality and rigor in the design and verification processes. Formal Methods (FMs) are a family of mathematically based methods for specification and verification which are characterized by a very high degree of formality. FMs have therefore been used actively in the design and verification of safety-critical systems in other industries, such as aerospace, automotive, railway, and nuclear for several decades. In fact, most industrial safety standards require the use of formal specification and verification for critical components. With the advent of autonomous systems, FMs have also been proposed as a promising approach to address some of the safety assurance challenges they introduce. This has resulted in active research on FMs applied to autonomous systems over the past decade (Luckcuck et al., 2019), such as using formal specification in simulation-based testing of self-driving cars (Tuncali et al., 2020), and using contract-based design for development and verification of aircraft systems (Nuzzo et al., 2015). Maritime autonomy is faced with the same challenges and needs, however, the maritime industry has not yet seen significant adoption of FMs. Investigating FMs for design and verification of autonomous vessel control systems can therefore address a significant research gap in the current state-of-the-art.

Another trend in verification of cyber-physical systems is the use of hybrid system analysis techniques. Hybrid systems are dynamic systems that exhibit both continuous and discrete behavior. Hybrid systems theory, therefore, enables modeling and formal analysis of the interactions between digital controllers and the continuous processes they control (Tabuada, 2009). Recent developments in the fields of FMs and hybrid systems are closely related. FMs have traditionally been applied to discrete systems and significant research has been conducted to extend FMs to hybrid systems to make them more applicable to cyber-physical systems.

1.2 Research Questions and Objectives

The main research objective of this thesis is to investigate how formal approaches to the design and verification of control systems can contribute to solving the safety assurance challenges for autonomous vessels. Because autonomous functionality is built on top of GNC systems, the safety and robustness of an autonomous vessel control system is highly dependent on the performance of the GNC systems. An additional research objective of this thesis is therefore to improve the control system performance of autonomous vessels through innovations at the GNC level.

To fulfill these objectives, the thesis aims to answer the following research questions (RQs):

- RQ1:** What are the main challenges for safety assurance of control systems for autonomous vessels?
- RQ2:** How can formal approaches for design and verification contribute to solving the identified challenges from RQ1?
- RQ3:** How can innovations at the GNC level contribute to improved control system performance for autonomous vessels?

1.3 Research Methodology

The main research methodology of this thesis has been case-based work, where generic methodologies have been developed, and then tested in a case study. The case studies have included physical full-scale experiments with NTNU's autonomous ferry prototype milliAmpere and physical model-scale experiments with the offshore supply vessel CyberShip III in the wave tank at NTNU's Marine Cybernetics Lab. In addition, simulation studies have been conducted with models of real vessels, including NTNU's autonomous passenger ferry, milliAmpere II and the remote-controlled short-sea shipping vessel Eidsvaag Pioneer, which is instrumented by Kongsberg Maritime.

A significant part of the research effort has also involved getting an overview of the state-of-the-art and relevant work in the field of FMs. This has mainly been achieved through structured queries in scientific databases, as well as completing a PhD course in FMs at NTNU's Department of Information Security and Communication Technology. Finally, an important research methodology has been the interaction with the industry partners, DNV and Kongsberg Maritime, through project meetings and practical workshops. This has given valuable insight into the research challenges the industry faces when developing and assuring autonomous vessel control systems, which in turn has guided the work of this thesis.

1.4 Contributions at a Glance

This thesis is organized as a collection of papers, hence the main contributions are the research papers developed during the doctoral work. The following gives an

overview of the publications included as part of the thesis and a summary of their contribution to the research objective.

A: Peer-reviewed journal paper

Automatic simulation-based testing of autonomous ships using Gaussian processes and temporal logic

Tobias Rye Torben, Jon Arne Glomsrud, Tom Arne Pedersen, Ingrid B. Utne and Asgeir J. Sørensen (2022).

Published in Journal of Risk and Reliability, 2022, pp. 1-21.

doi: 10.1177/1748006X211069277

Addresses **RQ1** and **RQ2**.

Contribution: A methodology for automatic simulation-based testing of control systems for autonomous vessels. The methodology uses the formal logic Signal Temporal Logic (STL) to specify formal requirements to test against and uses the STL robustness metric to evaluate simulations against requirements. Furthermore, the methodology uses a Gaussian process model to estimate the STL robustness over a scenario space, including its expected value and uncertainty. These estimates are used to adaptively guide the test case selection towards cases with low expected robustness or high uncertainty, leading to efficient coverage of the scenario space and fast identification of test cases that violated the test requirements.

B: Peer-reviewed journal paper

Towards Contract-based Verification for Autonomous Vessels

Tobias Rye Torben, Øyvind Smogeli, Jon Arne Glomsrud, Ingrid B. Utne and Asgeir J. Sørensen (2023).

Published in Ocean Engineering, Volume 270, 15 February 2023, 113685

Addresses **RQ1** and **RQ2**.

Contribution: This paper investigates the use of contract-based methods to address both design and verification challenges of control systems for autonomous vessels. The paper first presents a formal framework for specification of components and assume-guarantee contracts using the syntax of the Z3 automated theorem prover. Then, the paper proposes a methodology for contract-based verification using the formal framework. The methodology is divided into 4 steps: (1) Identification and modeling of the interactions between the autonomous vessel and its operative environment in order to define the top-level component and contract, (2) stepwise refinement of the top-level component into detailed sub-components and sub-contracts, (3) definition of test setups for simulation-based testing in order to verify that components meet their contract, and (4) applying a recursive procedure for contract-based system verification. The framework and methodology are demonstrated in a case study with an autonomous passenger ferry.

C: Peer-reviewed journal paper

Resetting observer design for linear time-varying systems with application to dynamic positioning of marine surface vessels

Tobias Rye Torben, Andrew R. Teel, Øivind K. Kjerstad, Emilie H. T. Wittemann and Roger Skjetne (2022).

Provisionally accepted for publication in IEEE Transactions on Control Systems Technology

Addresses **RQ2** and **RQ3**.

Contribution: A resetting observer for linear time-varying (LTV) systems with a formal proof of stability using hybrid dynamical systems theory. The motivation for the observer is better handling of unmodelled dynamics and reactivity to external disturbances without compromising steady-state performance. A reset is triggered if the output estimation error exceeds predefined bounds. The proposed observer uses a finite-time observer approach to calculate corrected state estimates after a reset is triggered. The finite-time observer equations are derived for LTV systems, and a method for calculating the state transition matrices online is presented. The observer equations are formulated in a hybrid dynamical systems framework, and sufficient conditions for uniform global pre-asymptotic stability are given. The method is applied to observer design for dynamic positioning of a marine surface vessel and numerical simulations as well as model-scale experiments of this application show improved transient performance compared to state-of-the-art observers.

D: Peer-reviewed conference paper

On Formal Methods for Design and Verification of Maritime Autonomous Surface Ships

Tobias Rye Torben, Øyvind Smogeli, Ingrid B. Utne and Asgeir J. Sørensen (2022).

Published in Proceedings of the 2022 World Maritime Technology Conference, pp. 253-262.

Addresses **RQ1** and **RQ2**.

Contribution: This paper aims to introduce FMs to the maritime industry and motivate their use in the design and verification of autonomous vessel control systems. The paper gives a high-level introduction to FMs by illustrating a typical formal development process and presenting the tools and methods used at the different stages of this process. Then, the paper discusses the current practice for verification of maritime control systems and several challenges and needs going towards MASS. The paper then demonstrates the application of FMs to the design and verification of autonomous vessels by three specific examples: Temporal logic specification of COLREG, contract-based design, and automated simulation-based testing. Finally, some limitations of FMs are discussed.

E: Peer-reviewed conference paper

Evolution of Safety in Marine Systems: From STPA to Automated Test Scenario Generation

Tom Arne Pedersen, Åse Neverlien, Jon Arne Glomsrud, Imran Ibrahim, Sigrid Marie Mo, Martin Rindarøy, Tobias Torben and Børge Rokseth (2022).

Published in Journal of Physics: Conference Series, volume 2311, no. 1, 012016

Addresses **RQ1** and **RQ2**.

Contribution: This paper follows up on the suggested future work in **Paper A**, where Systems Theoretic Process Analysis (STPA) is used to generate safety requirements in a structured way, and these requirements are specified using STL and used verified using the methodology of **Paper A**. The approach is demonstrated in a case study with a system for automated path-following and speed control for optimal line-setting in the line fishing vessel MS Geir.

Co-author contributions from the candidate: The candidate has contributed to the conceptualization of the paper, as the idea originates from suggested future work from the **Paper A**. The candidate has also contributed through development support, discussions and consultation on applying the methodology from **Paper A** in this paper. Finally, the candidate has contributed by revising the manuscript before submission.

F: Peer-reviewed journal paper

Development and testing of a risk-based control system for autonomous ships

Thomas Johansen, Simon Blindheim, Tobias Rye Torben, Ingrid Bouwer Utne, Tor Arne Johansen and Asgeir J. Sørensen (2022).

Accepted for publication in Reliability Engineering & System Safety

Addresses **RQ2**.

Contribution: This paper is a collaboration between the participants of the ORCAS project and combines contributions from each of the three PhD projects in a combined study. This includes a control system designed with risk-based decision-making capabilities to improve its intelligence and enhance the safe operation of autonomous systems. Specifically, the control system uses a Bayesian Belief Network (BBN), derived from the systems theoretic process analysis (STPA), as a foundation for an online risk model, which represents the operational risk for an autonomous ship. Further, an electronic navigational chart (ENC) module is used to provide the ship control system with an accurate description of the environment and conditions. The control system is verified against safety and performance requirements by use of the automated simulation-based testing methodology of **Paper A**. The case study object for the paper is the short-sea shipping vessel Eidsvaag Pioneer. The behavior of the risk-based control system is compared to the behavior of Eidsvaag Pioneer obtained from historical operational data.

Co-author contributions from the candidate: The candidate has contributed by developing the verification methodology, writing the parts on verification, and producing and visualizing the verification results. Moreover, the candidate has participated in the conceptualization of the paper as well as contributed through several revisions and discussions throughout the development of the paper.

G: Peer-reviewed journal paper

Control allocation for Double-ended Ferries with Full-scale Experimental Results.

Tobias Rye Torben , Astrid H. Brodtkorb and Asgeir J. Sørensen (2020).

Published in International Journal of Control, Automation and Systems, volume 18, no. 3, 556–63,

doi: [10.1007/s12555-019-0658-4](https://doi.org/10.1007/s12555-019-0658-4)

Addresses **RQ3**.

Contribution: A novel control allocation algorithm for double-ended ferries with symmetrical thruster configuration. The allocation problem is formulated using the extended thrust representation, resulting in a four-dimensional constrained optimization problem. Using the thrust configuration constraint, the optimization problem is reduced to a scalar bounded optimization problem, for which there exist fast and robust solvers. The paper proposes a cost function and optimization bounds such that the allocation algorithm supports the standard way of performing manual thruster control on ferries. The real-time performance of the proposed algorithm is demonstrated in a simulation study, and full-scale experiments with the autonomous ferry prototype milliAmpere.

The following research papers have been developed and published as part of the doctoral work, but are not included as part of this thesis:

- Peer-reviewed conference paper

milliAmpere: An Autonomous Ferry Prototype Edmund F. Brekke, Egil Eide, Bjørn-Olav H. Eriksen, Erik F. Wilthil, Morten Breivik, Even Skjellaug, Øystein K. Helgesen, Anastasios Lekkas, Andreas B. Martinsen, Emil H. Thyri, Tobias Rye Torben, Erik Veitch, Ole A. Alsos, Tor Arne Johansen (2022).

Published in Journal of Physics: Conference Series, volume 2311, no. 1, 012029

- Peer-reviewed conference paper

Control allocation for Double-ended Ferries with Full-scale Experimental Results.

Tobias Rye Torben , Astrid H. Brodtkorb and Asgeir J. Sørensen (2020).

Published in IFAC-PapersOnLine, volume 52, no. 21, 45-50,

doi: [10.1016/j.ifacol.2019.12.281](https://doi.org/10.1016/j.ifacol.2019.12.281)

1.5 Thesis Organization and Overview

The thesis is organized as a collection of papers. It is divided into two main parts:

Part I

Part I of the thesis aims to motivate and give context to the research papers as well as provide the necessary background theory. The thesis also aims to act as a minimal body of knowledge on topics related to formal design and verification of control systems for autonomous vessels to aid future maritime researchers that choose to explore this path. An outline of Part I follows next.

Chapter 2 gives an introduction to autonomous vessels and their control systems. Chapter 3 gives an introduction to safety assurance for autonomy, as well as an overview of methodologies for hazard analysis and risk assessment. The STPA methodology for hazard analysis is introduced in more detail. Chapter 4 gives an overview and taxonomy of methods for verification of cyber-physical systems. Chapter 5 gives an introduction and overview of FMs. In addition, specific methodologies which are used in the research papers are introduced in more detail. Chapter 6 introduces hybrid systems theory and gives a more detailed treatment of the hybrid systems framework used in **Paper C**. Chapter 7 gives a discussion of the results. The research questions of the thesis are answered by summarizing the findings from the research papers. Concluding remarks and suggestions for future work are given in Chapter 8.

Part II

Part II contains the research papers included as part of this thesis.

Chapter 2

Autonomous Vessels

2.1 What is Autonomy?

Establishing a precise definition for autonomous vessels is not as straightforward as one might think, and there have been several attempts at definitions and taxonomies (Rødseth et al., 2022). Multiple characteristics relate to what's conceived as autonomous. The term autonomy indicates some level of self-governance or independent operation. However, independent operation is not a sufficient characteristic for a vessel to be considered autonomous. Traditional *automation* systems, such as dynamic positioning (DP), can operate independently, but DP vessels certainly are not considered autonomous. For a vessel to be considered autonomous, there also is a need for increased mission complexity. While an automation system can perform well-defined tasks independently, an autonomous system must perform complex tasks under significant uncertainty in an unstructured environment (Sørensen and Ludvigsen, 2018). To achieve this, autonomous systems must be *deliberative*, meaning that they can model and comprehend the external and internal situation and plan their actions accordingly, hence making deliberate choices. A third characteristic that causes confusion around the definition of autonomy is the manning level. The terms unmanned and autonomous vessels are often used interchangeably. However, they are not the same. For instance, remote-controlled vessels can be completely unmanned but not autonomous. Similarly, an autonomous vessel can operate fully independently and have a crew onboard for monitoring the autonomy or performing other tasks.

The International Maritime Authority (IMO) has established the term Maritime Autonomous Surface Ship (MASS), which they define as “*a ship which, to a varying degree, can operate independently of human interaction*”. This broad definition includes both manned and unmanned ships and the autonomous functionality can range from decision support systems to full autonomy. To give a more refined taxonomy, IMO also defines levels of autonomy:

Level 1 **Ship with automated processes and decision support:** Seafarers are on board to operate and control shipboard systems and functions. Some operations may be automated and at times be unsupervised but with seafarers

on board ready to take control.

Level 2 **Remotely controlled ship with seafarers on board:** The ship is controlled and operated from another location. Seafarers are available on board to take control and operate the shipboard systems and functions.

Level 3 **Remotely controlled ship without seafarers on board:** The ship is controlled and operated from another location. There are no seafarers on board.

Level 4 **Fully autonomous ship:** The operating system of the ship is able to make decisions and determine actions by itself.

This is only one of many attempts at defining levels of autonomy.

To better understand the concept of autonomy, it can be informative to look at the feedback loops that exist in modern vessels, and how this picture changes for autonomous vessels. Figure 2.1 shows the nested feedback loops for a typical modern vessel. The propulsion system in the innermost loop is responsible for managing power production and distribution and controlling the thrusters to actuate the vessel in accordance with a specified propulsion setpoint. The middle feedback loop is responsible for the motion control of the vessel. A navigation system estimates the motion of the vessel based on sensor data, and automatic motion controllers, such as autopilot and DP, produce propulsion setpoints to steer the vessel toward the specified motion setpoints. The outermost feedback loop is manual. The lookout crew senses the operative environment of the vessel to obtain situational awareness. The situational awareness is distributed to the captain or helmsman, which produces motion setpoints in order to achieve the mission goals, such as completing a voyage.

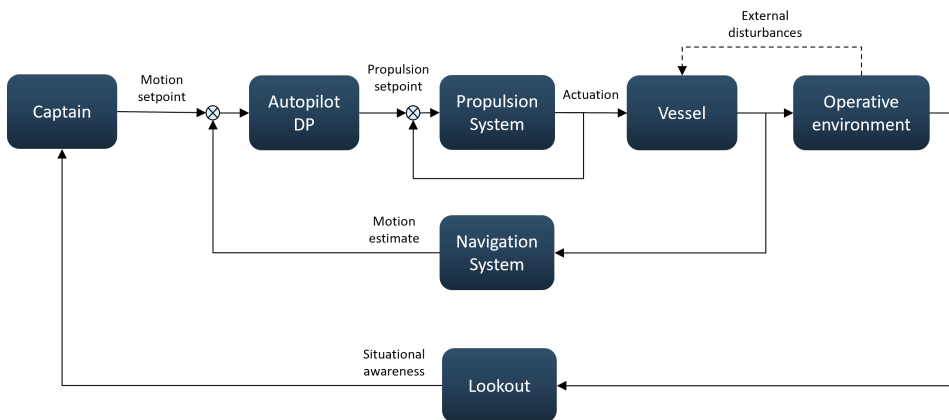


Figure 2.1: The nested feedback loops in a conventional vessel. The outermost loop is manual and performed by the onboard crew.

Figure 2.2 shows the feedback loops for an autonomous vessel. What has changed here, is that the outermost loop is automated, such that the human crew can be

moved outside the real-time control loop. This is achieved by adding a situational awareness systems, which senses the environment using perception sensors and creates a situational awareness model. This model is given to a high-level controller, which makes deliberate decisions and motion plans in order to achieve a specified mission goal.

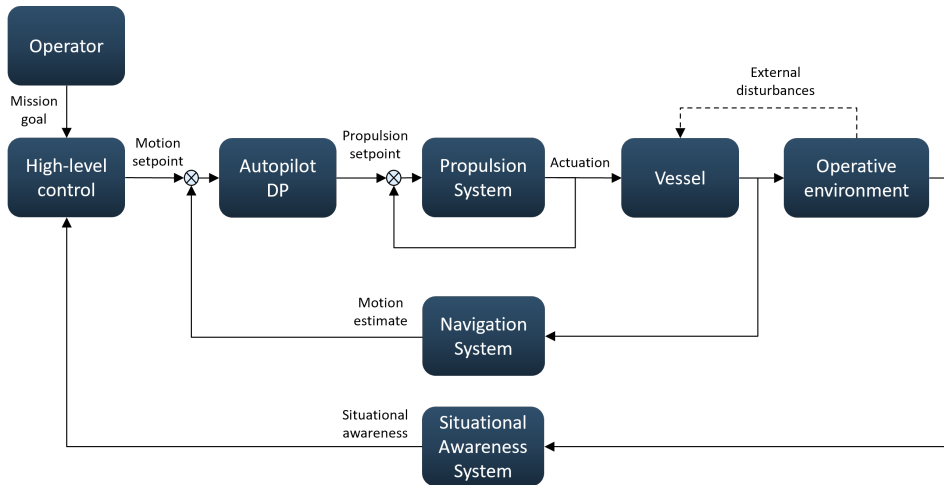


Figure 2.2: The nested feedback loops in an autonomous vessel. The outer loop is automated by an autonomy system, enabling the human operator to be removed from the real-time control loop.

2.2 Control System Design

Next, an overview of the inner workings of autonomous vessel control systems is given. A typical functional architecture is shown in Figure 2.3. Although the grouping of functions may vary between specific autonomous vessel designs, this architecture contains the main elements that are present in state-of-the-art autonomous vessel control designs. The modules in Figure 2.3 are generally arranged left-to-right according to their place in the sense-model-plan-act process. Modules are arranged top-to-bottom according to their place in the control hierarchy, such that upper modules provide control objectives to lower modules, and lower modules provide feedback to upper modules.

2.2.1 Operator Interface

At the top of the control, hierarchy is the human operator, which interacts with the autonomy system through an operator interface (Veitch and Alsos, 2022). This interface may contain remote control functionality and display monitoring information and alarms such that an operator can take action in case of unexpected

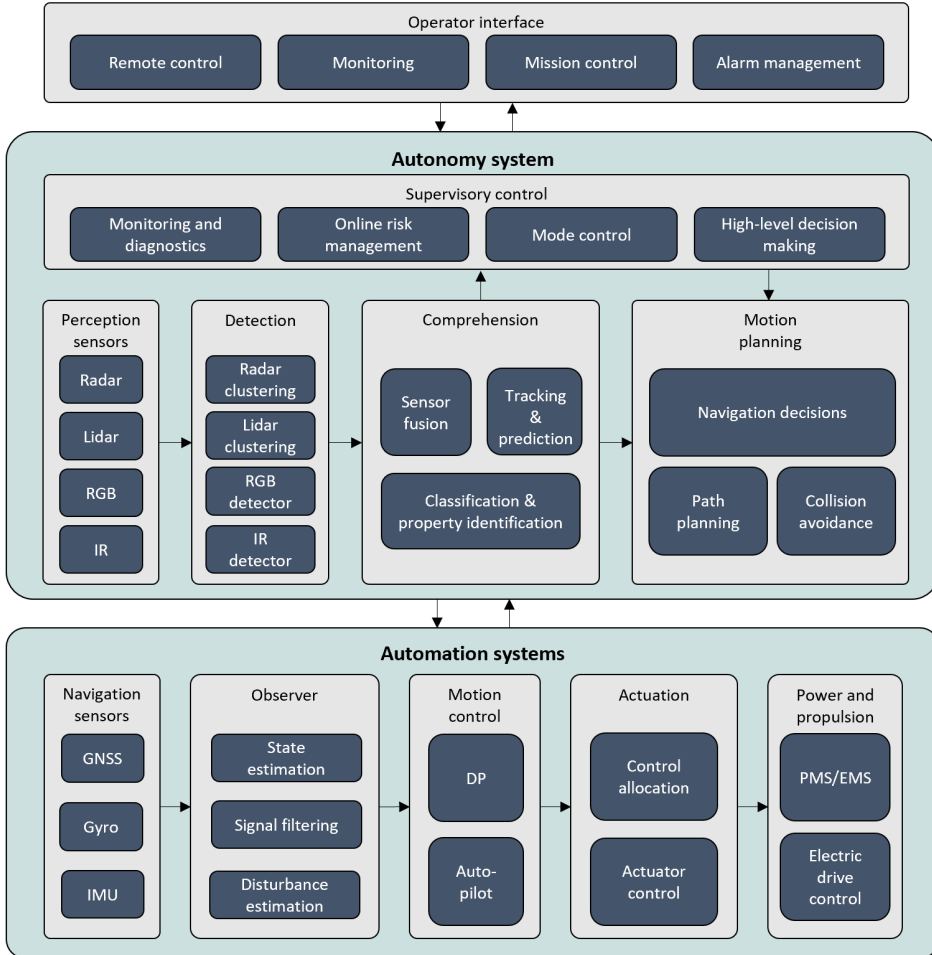


Figure 2.3: Functional architecture for autonomous vessel control systems.

situations. The operator may also provide the autonomy system with high-level control objectives (missions), such as starting a voyage with a specified destination.

2.2.2 Autonomy System

One level down in the control hierarchy is the autonomy system, which corresponds to the outer feedback loop in Figure 2.2. The autonomy system senses its surroundings using perception sensors, such as radars, lidars, RGB cameras, and infrared (IR) cameras. The raw, high-dimensional sensor data are processed by a detection module, which uses clustering and detection algorithms to identify static and moving obstacles (Thombre et al., 2022). The detections from the various sensors are processed by a comprehension module, which uses sensor fusion algorithms

to merge the detections from the individual sensors into fused object detections. Furthermore, the comprehension module tracks the detection over time to estimate the motion of obstacles and predict their future motion (Wilthil et al., 2017). The comprehension module may also classify the type of objects, for instance, to distinguish between power-driven vessels and sailing vessels. It may also estimate properties of objects, for instance, extended object tracking can be used to estimate the size of objects. Together, the perception sensors, detection, and comprehension modules comprise the situational awareness system of an autonomous vessel. Note that Figure 2.3 shows a *detect-and-fuse* pipeline. The alternative *fuse-and-detect* pipeline is also a possibility, both alternatives have their merits and caveats.

The motion planning module is responsible for producing a motion reference in order to achieve a specified mission goal while avoiding the objects tracked by the situational awareness system (Vagale et al., 2021). This may include several functions, such as long-term path planning to optimize energy consumption and voyage time, and short-term collision avoidance. The motion planning makes navigational decisions based on e.g electronic navigational charts (ENCs), COLREG, and sea marks. The motion planning module may also include functionality for auto-docking. The motion reference produced by the motion planning module is typically in form of waypoints or a full-state trajectory. The motion reference is outputted to the automation systems, which is tasked with tracking this reference.

A third module in the autonomy system is the supervisory control module. Supervisory control is responsible for the overall coordination of the autonomy system to achieve the mission goal and ensure the safety and integrity of the system (Johansen and Utne, 2022). Supervisory control may include functionality such as high-level decision-making, controlling the operational modes, and monitoring and diagnostics of the vessel systems (Blanke et al., 2000). The high-level decision-making may incorporate online risk management functionality, where the decisions are based on online risk models in order to achieve risk-aware behavior. This is the topic of **Paper F**.

2.2.3 Automation Systems

At the lowest level of the control hierarchy are the automation systems, which are commonplace on conventional DP vessels. The automation systems correspond to the middle and inner feedback loops in Figure 2.1 and include GNC functionality in addition to monitoring and control of power and propulsion systems. In the control hierarchy, automation systems are responsible for powering and controlling the actuators such that the vessel tracks the motion reference received from the autonomy system. The motion of the vessel is measured using navigation sensors such as Global Navigation Satellite System (GNSS) receivers to measure position, gyro compasses to measure heading, and inertial measurement units (IMUs) to measure accelerations and angular velocities. The measurements from the navigation sensors are input to the *observer*, which combines all measurements with a vessel model to produce smooth full-state state estimates. The observer may

also include filtering functionality, such as removing high-frequency measurement noise and wave-frequency signal components (Fossen et al., 1999). Measurement integrity functionality such as wildpoint filtering, sensor weighting, and sensor voting are typically also included here for robustness. The observer can also include disturbance estimation functionality, which can be used to achieve better disturbance rejection in the motion control loop. Observer design is the topic of **Paper C**.

The motion control module is a feedback controller which compares the motion estimates from the observer with the motion reference from the autonomy system and produces a control action using e.g. PID control in order to track the reference. Examples of motion controllers are DP controllers (Sørensen, 2011), that perform trajectory tracking in surge, sway, and heading, and autopilots, that perform path following by controlling the speed and heading (Fossen et al., 2003). The output from the motion control module is a control action in the form of a commanded force in surge, sway, and yaw. The actuation module is responsible for coordinating and controlling the actuators in order to produce the specified force. The control allocation module distributes the control force commanded by the motion control module to the individual actuators. The actuators may include azimuth thrusters, tunnel thrusters, main propellers, and rudders. Control allocation is usually solved as an optimization problem, where the optimization objective is to produce the specified control force as accurately as possible while using a minimal amount of energy, reducing wear and tear, and respecting actuator constraints, such as thruster saturation limits and actuator angle limits (Johansen and Fossen, 2013). Control allocation is the topic of **Paper G**. The actuator control module controls the speed and angle of the individual actuators in order to produce the thrust specified by the control allocation module. This can for instance include servo control of the angle for azimuth thrusters and rudders and torque control of the speed for thrusters (Martelli and Figari, 2022).

Finally, the power and propulsion system is responsible for producing and distributing power to the actuators (Radan, 2008). This can include a power management system (PMS), which includes functions such as automatic starting and stopping of generators, load shedding, detection and isolation of electrical failures, and blackout prevention. For vessels with energy storage systems, such as batteries, the power and propulsion module may also have an energy management system (EMS) for managing the charge and discharge of energy storage devices. The interface between the digital control system and the physical power system is typically the electric drives for the actuators, where the control signals control the power through the electric drives to achieve the desired motor speeds and actuator angles.

Chapter 3

Safety Assurance

This chapter first gives a brief introduction to safety assurance and safety assurance concepts for autonomous systems. Then, an overview of methodologies for hazard analysis and risk assessment is given, with a more detailed introduction to STPA, which is used in several of the research papers in this thesis.

3.1 Safety Assurance of Autonomy

Assurance can be defined as *ground for justified confidence*. Hence, safety assurance is the process of establishing ground for justified confidence that a system is sufficiently safe. The level of required confidence on the criticality of the system (DNV, 2018). The safety assurance process can be split into two main steps:

- Establish a sufficient set of safety requirements
- Produce verification evidence to show that the system meets the safety requirements

Verification is the topic of Chapter 4, however, if the safety requirements are not sufficiently complete, the system will be unsafe despite rigorous verification efforts. Autonomous systems represent a fundamental change in this regard, as it is very difficult to establish a sufficiently complete set of safety requirements due to their extensive sensing and interaction with the open environment. The safety assurance challenge for autonomous systems has led to an evolution in the tools and methodologies for safety assurance. The automotive industry has been a key driver in this development (Koopman et al., 2019). Because the automotive industry is years ahead of the maritime industry in terms of autonomy, their approach to safety assurance will likely be highly influential on the maritime industry. Moreover, the maritime industry does not have established safety standards for autonomy. Safety assurance concepts are therefore exemplified by automotive safety standards in the following.

Traditionally, systems have been designed with the philosophy that the human operator is ultimately responsible for safe operation. The safety assurance process has therefore focused on mitigating known hazards caused by equipment failure. This philosophy is referred to *functional safety*. For the automotive industry, required

activities for functional safety are described in the ISO 26262 safety standard (ISO, 2018). While functional safety lays the foundation for developing a safe system, it is not sufficient when the level of autonomy increases. The automotive industry has developed complementary safety standards to address this.

As complex tasks are moved from the human operator to the control system, there may occur situations where a system demonstrates unsafe behavior even though no equipment failure has occurred. This renders functional the safety approach insufficient. Such unsafe situations can for instance be caused by insufficient situational awareness, unexpected environmental interactions, or gaps in the requirements. The automotive industry has developed the *safety of the intended functionality* (SOTIF) concept to address such aspects. SOTIF is covered by the automotive safety standard ISO/PAS 21448 (ISO/PAS, 2022).

A third safety standard for automotive autonomy is UL 4600 (UL, 2022), which proposes the use of a *safety case* as the overarching structure of the safety assurance process. A safety case is a structured argument, supported by evidence, intended to justify that a system is acceptably safe for a specific application in a specific operating environment. UL 4600 aims to combine aspects from ISO 26262 and ISO/PAS 21448 with additional aspects for highly autonomous vehicles in a coherent safety argument structured as a safety case.

3.2 Hazard Analysis and Risk Assessment

Hazard analysis and risk assessment, commonly referred to as HARA, are key activities in the safety assurance process. HARA provides means for systematic identification of hazards, and can therefore be instrumental in achieving sufficient completeness in the safety requirements.

A common approach is to use Preliminary Hazard Analysis (PHA), also known as HAZID, at an early design stage to get an overview of the overall risk picture. Later in the development process, more detailed analyses are conducted. Some methodologies that have traditionally been used include Failure Mode and Effect Analysis (FMEA), Fault Tree Analysis (FTA), and Event Tree Analysis (ETA). These have been successfully applied in hardware-dominated systems where the software is limited to simple functions. However, they provide little aid in identifying hazards for more complex and software-intensive systems. The use of Systems Theoretic Process Analysis (STPA) has been proposed as a candidate for complex maritime systems (Rokseth, 2018) and autonomous vessels in particular (Rokseth et al., 2019). STPA is a key methodology in **Paper E** and **Paper F** and is also discussed superficially in **Paper A** and **Paper B**. A brief introduction to STPA is therefore given next.

The key idea behind STPA is to treat system safety as a dynamic control problem instead of focusing only on preventing and containing failures (Leveson, 2011). The

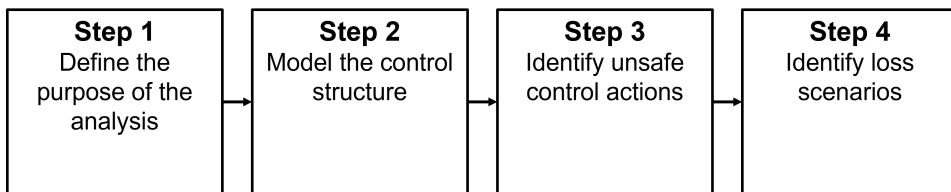


Figure 3.1: The four steps of an STPA. Figure redrawn based on similar figure from Leveson and Thomas (2018).

four steps of an STPA are shown in Figure 3.1.

Step 1 of the analysis starts by defining the system, system boundary, and environment. Then, *losses* are identified. A loss is defined as an outcome that is unacceptable to stakeholders. *System-level hazards* that can lead to the identified losses are identified next. A system-level hazard is defined as a system state that, together with a set of worst-case environmental conditions, will lead to a loss. Finally, *system-level constraints* are defined in order to avoid system-level hazards.

Step 2 of the analysis involves modeling the system as a hierarchical control structure. The system is modeled as a set of interacting controllers and controlled processes. Controllers send control actions to the controlled processes and the controlled processes send feedback to the controllers. Responsibilities are assigned to the controllers by mapping system-level constraints onto them. Specific control actions which implement the responsibilities of the controllers are identified.

Step 3 of the analysis involves identifying *unsafe control actions* (UCAs). A UCA is defined as a control action that, in a particular context and worst-case environment, will lead to a hazard. UCAs are systematically identified by checking each control action against the following patterns:

1. Not providing the control action leads to a hazard.
2. Providing the control action leads to a hazard.
3. Providing a potentially safe control action but too early, too late, or in the wrong order leads to a hazard.
4. Providing the control action for too long or too short leads to a hazard.

Based on the identified UCAs, controller constraints are specified, which are safety requirements to prevent UCAs.

Finally, Step 4 of the analysis involves identifying *loss scenarios*, that describe causal factors that can lead to UCAs or directly to hazards.

Chapter 4

Verification

Verification is a loaded term whose precise definition varies with the context of use. Even within the context of verification of cyber-physical and embedded systems, there does not seem to be agreement on the precise meaning of verification. Moreover, the terms verification, validation, and testing are often used interchangeably. In this thesis, verification refers to the process of producing evidence that a system meets a specified set of requirements. Testing is one possible verification technique. This chapter gives an overview of verification methods for cyber-physical systems.

4.1 Verification of Cyber-physical Systems

Cyber-physical systems are comprised of interacting physical and digital components. A typical configuration is that a physical process is controlled by digital controllers, which is the case for autonomous vessels. The interfaces between the physical and digital worlds are the sensors and actuators. Analog-to-digital (A/D) converters convert the analog sensor readings to digital signals, and digital-to-analog (D/A) converters convert digital control signals to analog signals that drive the actuators. The dual nature of cyber-physical systems introduces challenges for verification of them, especially as their complexity increases (Kapinski et al., 2016). Verification of cyber-physical systems has therefore been an active area of research and development. Next, an overview of methodologies for verification of cyber-physical systems is given.

Figure 4.1, shows a classification of prominent verification methods. The methods are placed on a spectrum that ranks their scalability, that is, how large or complex systems they can verify, and their exhaustiveness, which is an indication of how thoroughly the possible behaviors of a system are covered. Figure 4.2 shows the corresponding models used in the verification. The models are ranked according to their fidelity, that is, how well their behavior matches the behavior of the real implementation that they model. The methods are classified into three distinct classes; simulation-based, formal, and analytical. An introduction to these classes of verification methods is given next.

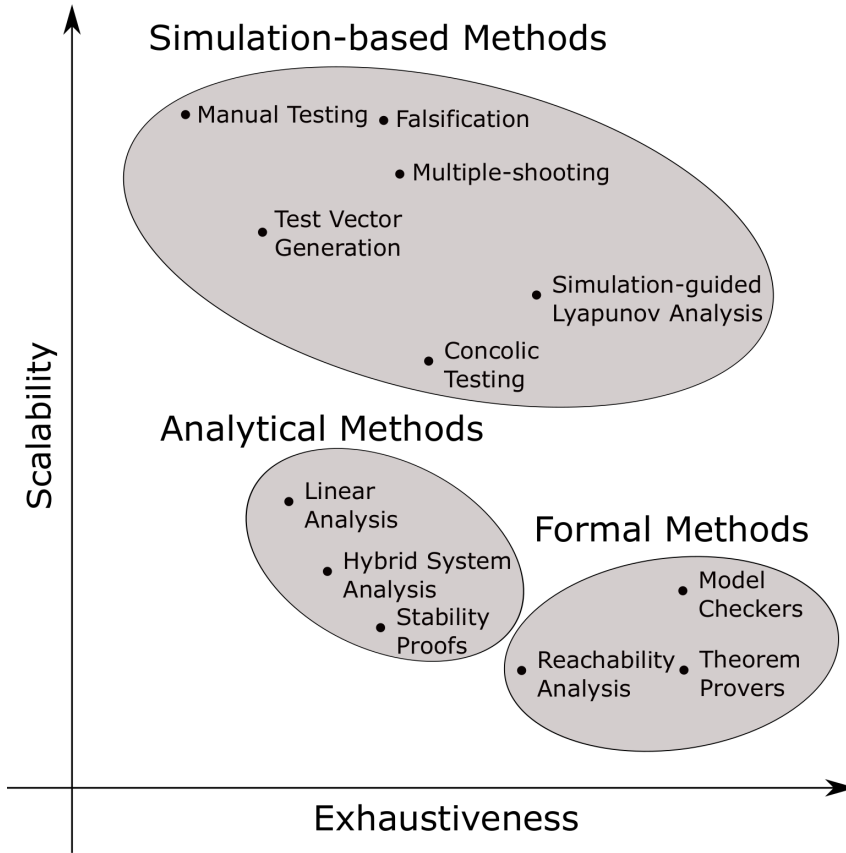


Figure 4.1: A spectrum of verification methods for cyber-physical systems. The horizontal axis indicates the level of exhaustiveness. The vertical axis indicates the scalability of the method. Source: **Paper A**.

4.2 Simulation-based Verification

A simulation model is a mathematical model, such as a differential equation, which can be solved approximately using numerical solvers. Given an initial condition and an input signal, a simulation model can approximate the time-domain behavior of the modeled system. Simulation-based verification methods use simulation models to analyze the behavior of cyber-physical systems in order to verify that their behavior meets the requirements. Figure 4.2 divides simulation models into three classes; Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), and Hardware-in-the-Loop (HiL). In a MiL simulation, both the controller and the physical process are simulation models. In a SiL simulation, the physical process is controlled by the real control software. In a HiL simulation, the control system runs on the real hardware platform and communicates with the simulated physical process through a HiL interface. It is possible to create detailed and realistic simulation models of

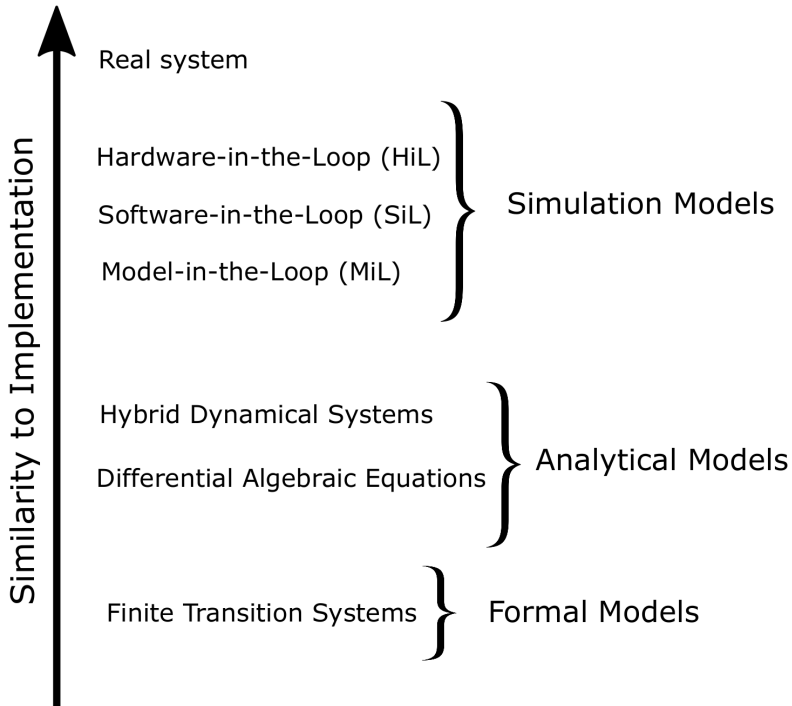


Figure 4.2: Spectrum of the models used by the verification methods from Figure 4.1. The models are ranked according to their fidelity, that is, how closely their behavior resembles the behavior of the real implementation that they model. Source: **Paper A**.

large and complex cyber-physical systems, even entire ships including the 3D virtual environment. Together with the HiL and SiL opportunities, simulation-based verification is both highly scalable and the models can achieve high fidelity.

Simulation is inherently a testing approach, as a single simulation only assesses a single behavior. Manual simulation-based testing, therefore, scores low on exhaustiveness in Figure 4.1. The other simulation-based methods in the figure represent different approaches to improve the exhaustiveness of simulation-based verification by systematically managing the test case selection and evaluation of the results. Test vector generation is an automated process for selecting test cases such that certain coverage criteria are met. Concolic testing combines simulations of the physical process with a formal analysis of the decision branching of the controller software. Falsification methods use optimization approaches or reinforcement learning to search for behaviors that violate requirements. Multiple-shooting approaches attempt to achieve falsification by running many partial simulations and splicing together the results to identify a falsifying case. Simulation-guided Lyapunov analysis refers to a method for searching for a Lyapunov function using the results of

simulations. Stability, invariant sets, and performance bounds can be derived from the Lyapunov function. A more detailed treatment of this topic is given in Kapinski et al. (2016).

4.3 Formal Verification

FMs are characterized by having a high level of formality and exhaustiveness, as Figure 4.1 indicates. The models used by FMs are usually some form of a finite transition system. These are discrete-time models containing a finite number of states and a finite number of transitions between states. For such models, many formal verification techniques can achieve fully exhaustive verification, where all possible behaviors are verified against the requirements. However, FMs typically suffer from limited scalability due to, among others, being limited by the so-called *state-space explosion* problem. This refers to the fact that the number of states grows exponentially as the number of variables in a system increases. For cyber-physical systems, formal verification techniques can also suffer from the *reality gap* problem, because the finite-state models are abstractions of the real system, which may cause a significant gap between the model being verified and the implemented system (Luckcuck et al., 2019). The formal models are therefore placed low on the fidelity spectrum of Figure 4.2. FMs are described in more detail in Chapter 5.

4.4 Analytical Verification

Analytical verification methods use manual mathematical analysis of symbolic equations. For continuous systems, the models are typically some form of Differential Algebraic Equation (DAE). Systems that exhibit both continuous and discrete behaviors can be modeled as hybrid systems. Hybrid systems are of particular interest for cyber-physical systems, which are inherently hybrid. Both DAEs and hybrid system models rank quite low on fidelity in Figure 4.2 due to the sheer difficulty in creating realistic analytical models of complex systems. In Figure 4.1, Stability proofs refer to using mathematical analysis, such as Lyapunov methods to prove e.g. stability or invariance of a set for continuous systems (Khalil, 2002). Linear analysis is similar, but here the system is first linearized about an operational point. This is quite trivial even for complex models however, the results are only local and thus not very exhaustive (Chen, 1999). Linear analysis is also possible to do numerically for some simulation models. Hybrid system analysis refers to a set of mathematics studying the properties of hybrid dynamical systems (Goebel et al., 2012). Hybrid system analysis is described in more detail in Chapter 6.

Chapter 5

Formal Methods

This chapter first gives a brief introduction and overview of the field of FMs. Then, it provides a more detailed theoretical background on the specific FMs which are applied in the research papers of this thesis.

5.1 Introduction and Overview

FMs can be defined as a family of mathematically based methods for specification and verification (Woodcock et al., 2009). The theoretical foundations for FMs originate mainly from computer science and discrete mathematics. Early development of FMs dates to the 1960s, where they were first applied to the design of logic circuits and primitive computer programs.

The most attractive trait of FMs is that they offer a high level of assurance of the safety and correctness of the systems they are applied to. FMs have therefore seen significant adoption in safety-critical applications in industries such as aerospace, automotive, railway, and nuclear (Lecomte et al., 1991). In fact, most functional safety standards, such as IEC 61508 (IEC, 2010) and ISO 26262 (ISO, 2018), have formal specification and verification as mandatory activities for critical components. FMs have also been proposed as a candidate to address some of the assurance challenges of autonomous systems (Luckcuck et al., 2019).

The maritime industry does not have a tradition of using FMs. However, in addition to the publications of this thesis, a few publications have emerged recently on FMs for autonomous vessels. Lu et al. (2016) and Yan et al. (2018) propose formal models of COLREG-based collision avoidance that uses model checkers to analyze safety properties. Shokri-Manninen et al. (2020) have created a formal automata-based model of single-vessel encounters and synthesized a correct-by-construction navigation strategy. Meyer et al. (2020) and Park and Kim (2020) have synthesized correct-by-construction controllers for automatic docking of marine vessels based on reachability analysis. Foster et al. (2020) present a controller for autonomous marine vessels in form of a hybrid dynamical system and use an automated theorem prover to verify some safety invariants. Krasowski and Althoff (2021) propose

a temporal logic formalization of the COLREG.

Figure 5.1 shows the main steps and activities in a formal development process. In the following, this figure is used to introduce the different FM tools and show where they fit in the development process.

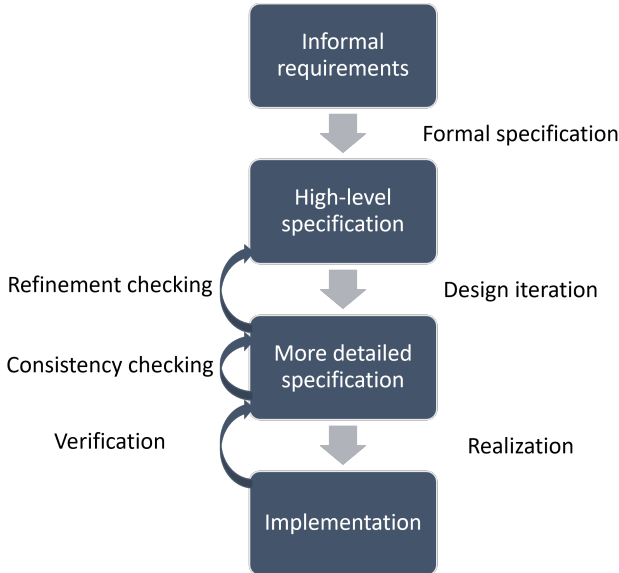


Figure 5.1: Steps and activities in a formal development process. From **Paper D**.

The first step in most development processes is the specification of requirements. Requirements are typically formulated informally using natural language, possibly augmented with some mathematical formulas. Formal specification involves taking the informal requirements and specifying them using a *formal specification language*. Such languages have clearly defined syntax and semantics, that is, there are strict rules on valid statements, and it is clearly and unambiguously defined how the statements are to be interpreted. This not only means that they can be subject to mathematical analysis, but also that they are machine-readable and, therefore, can be subject to automated reasoning by a computer. Examples of formal specification are temporal logics, described in Section 5.3, and assume-guarantee contracts, described in Section 5.4.

After high-level specification, the next step in a formal development process is typically to stepwisely refine the specification in a series of design iterations. Each design iteration adds more detail to the specification and brings it closer to something which can be realized in hardware and software. An important activity for each design iteration is refinement checking. Refinement and refinement checking of assume-guarantee contracts is described in Section 5.4 and used extensively in **Paper B**. Another important step in the design iterations is consistency checking.

This involves checking a specification for contradictions.

When the formal specification is sufficiently detailed, the next step is to realize the specification in the form of a hardware or software implementation. This may simply involve manual programming from the specification. Although this does not utilize the full potential of a formal development process, it is easier to produce correct code when writing from a clear and unambiguous specification where fundamental design flaws have been removed at the design stages. For some specification languages, there also exist tools that can automatically generate a correct-by-construction implementation from the specification. Examples of this include generation of computer code in various programming languages and the synthesis of controllers which control a system to meet the specification (Meyer et al., 2020).

The final step in the formal development process is formal verification. The most widespread formal verification technique is model checking, which exhaustively checks that all possible executions of the system are in accordance with the formal specification. Some prominent model checking tools are SPIN (Holzmann, 1997), NuSMV (Cimatti et al., 2002) and UPPAAL (Larsen et al., 1997). Another important class of formal verification techniques is automated theorem provers. These are used in **Paper B** and will be described in more detail in Section 5.5. Reachability analysis is another formal verification technique that can compute a set of reachable states from any state in a system. Reachability analysis can, for instance, be used to formally verify safety by showing that some unsafe set is not reachable from any state (Meyer et al., 2021).

5.2 Preliminaries

Before the specific FMs used in this thesis are introduced, some mathematical preliminaries relevant to all of them are presented.

5.2.1 Propositional Logic

Propositional logic is a fundamental theory in the field of FMs, and propositional logic notation and concepts are used heavily throughout the remainder of this chapter as well as in the research papers. The main concepts and notation are therefore introduced here. The reader is referred to Huth and Mark (2004) for a complete treatment of the topic.

Propositional logic studies the logical relationships of statements connected via logical operators. A *statement* is a declarative sentence that can be either true or false, but not both. Statements can therefore be considered Boolean variables. An example of a statement is given below.

$$p = \text{"The main engine is running"} \tag{5.1}$$

Propositional logic formulas are constructed by combining statements with logic operators. Consider two statements p and q . The operators of propositional logic are given below, with their natural language translation in parentheses.

- *Negation*: $\neg p$ (not p)
- *Conjunction*: $p \wedge q$ (p and q)
- *Disjunction*: $p \vee q$ (p or q)
- *Implication*: $p \rightarrow q$ (if p , then q)

The semantics of these operators can be completely specified in form of a truth table, as shown in Table 5.1. The true value is denoted \top , and the false value is denoted \perp .

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \rightarrow q$
\top	\top	\perp	\top	\top	\top
\top	\perp	\perp	\perp	\top	\perp
\perp	\top	\top	\perp	\top	\top
\perp	\perp	\top	\perp	\perp	\top

Table 5.1: Truth table for propositional logic operators.

5.2.2 States, Behaviors, and Signals

Next, some general definitions of *states*, *behaviors*, and *signals* are given. Note that different methodologies often use different terms and notations to describe the same concepts. The notation used in the research papers, therefore, differs slightly. Here, unified definitions are given, which are used in the remainder of this chapter to clearly show the connection between the methodologies.

Consider a system that at any time can be described by the values of the variables v_1, v_2, \dots, v_n , each with domain D_1, D_2, \dots, D_n . The *state* s of the system is defined as the vector collecting all variables, such that $s = [v_1, v_2, \dots, v_n] \in \mathcal{S}$, where $\mathcal{S} = D_1 \times D_2 \times \dots \times D_n$. A *behavior*, σ , of the system is defined as a sequence of states over time, such that $\sigma = s_0, s_1, s_2, \dots, s_N$ for $N \in \mathbb{N}$. Furthermore, let $t = [t_0, t_1, t_2, \dots, t_N]$ denote the timestamps corresponding to the states in σ , such that $s = s_0$ at t_0 , $s = s_1$ at $t = t_1$ and so forth. Finally, a *signal* w is defined as the collected sequence of state-time pairs $w = (s_0, t_0), (s_1, t_1), \dots, (s_N, t_N)$.

5.3 Temporal Logic

Temporal logics are a set of specification languages that are particularly well-suited for specifying temporal aspects of behaviors. Classical propositional logic can express static relationships using the logic operators introduced in Section 5.2.1. Temporal logic extends propositional logic by adding temporal operators, such as *always* (\square), *next* (\bigcirc), *eventually* (\diamond) and *until* (U). A temporal logic formula specifies a set of behaviors that satisfy the formula, and there exist effective algorithms

to check whether a given behavior satisfies a temporal logic formula or not. The original form of temporal logic, Linear Temporal Logic (LTL), was introduced in Pnueli (1977). LTL operates on Boolean behaviors in discrete time and is extensively used to specify behaviors of finite-state transition systems for model checking. An extension of LTL is Metric Temporal Logic (MTL) which operates on Boolean behaviors in continuous time (Alur and Henzinger, 1993). Signal Temporal Logic (STL) was proposed as a syntactic addition to MTL, where the formulas operate on real-valued behaviors (Maler and Nickovic, 2004). Because STL can specify continuous-time, real-valued behavior, it is particularly well-suited for specification of cyber-physical systems, and it is the form that is used in **Paper A**.

5.3.1 Signal Temporal Logic

The basic building block of STL formulas is *predicates*. A predicate π is a function that maps a state $s \in \mathcal{S}$ to a Boolean. In STL, the predicates take the form

$$\pi ::= f(s) \leq c, \quad (5.2)$$

where f is a scalar, real-valued function which maps the input s to a real-valued scalar, and c is a real-valued scalar.

STL formulas are built by combining predicates with the operators of propositional logic and temporal operators. The temporal operators can also be specified over specific time intervals. An informal description of the temporal operators is given before the formal syntax and semantics of STL are presented.

- *Eventually*: $\diamond\varphi$ is true if φ is true at some time.
- *Always*: $\square\varphi$ is true if φ is true at all times.
- *Next*: $\bigcirc\varphi$ is true if φ is true at the next discrete time step.
- *Until*: $\varphi_1 U \varphi_2$ is true if φ_1 is true until φ_2 first becomes true.
- *Release*: $\varphi_1 R \varphi_2$ is true if φ_2 is true until φ_1 first becomes true.

Definition 5.1. STL Syntax (Maler and Nickovic, 2004):

Let Π be the set of predicates and \mathcal{I} be any non-empty connected time interval of $\mathbb{R}_{\geq 0}$. The set of well-formed STL formulas is defined by the grammar

$$\varphi ::= \top \mid \pi \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 U_{\mathcal{I}}\varphi_2, \quad (5.3)$$

where φ , φ_1 and φ_2 are STL formulas, \top is the True constant and π is a predicate.

Note that all the logical and temporal operators can be derived from these basic operators:

$$\varphi_1 \wedge \varphi_2 \equiv \neg(\neg\varphi_1 \vee \neg\varphi_2) \quad (5.4)$$

$$\varphi_1 \rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2 \quad (5.5)$$

$$\diamond_{\mathcal{I}}\varphi \equiv \top U_{\mathcal{I}}\varphi \quad (5.6)$$

$$\square_{\mathcal{I}}\varphi \equiv \neg\diamond_{\mathcal{I}}\neg\varphi \quad (5.7)$$

$$\varphi_1 R_{\mathcal{I}}\varphi_2 \equiv \neg(\neg\varphi_1 U_{\mathcal{I}}\neg\varphi_2) \quad (5.8)$$

The following is an example of a simple STL formula that specifies that a vessel must always keep a safe distance from other vessels:

$$\varphi_{safety} = \Box \neg (d(t) \leq d_{min}) \quad (5.9)$$

where $d(t)$ is the distance to another vessel at time t and d_{min} is a constant specifying the minimum required safety distance.

5.3.2 STL Robustness Metric

An attractive property of STL is that there exists quantitative semantics, called the *STL robustness metric* (Fainekos and Pappas, 2009). The normal Boolean semantics only give a true/false evaluation of whether a behavior satisfies a formula. The STL robustness metric gives a quantitative evaluation of how robustly a behavior satisfies an STL formula. This is a powerful combination, as STL both provides a language to specify behaviors and a metric to measure conformance to these behaviors.

Let $\llbracket \varphi \rrbracket(w, t_i)$ denote the STL robustness of the signal w against the formula φ at time t_i . The STL robustness metric has the property that $\llbracket \varphi \rrbracket(w, t_i) \geq 0$ if w satisfies φ at time t_i . Otherwise, $\llbracket \varphi \rrbracket(w, t_i) < 0$. Informally, the magnitude of the robustness metric indicates how much the behavior can change without violating the requirement. Before the STL robustness metric is formally defined, the *signed distance* to a set is defined.

Definition 5.2. Signed distance (Fainekos and Pappas, 2009):

Consider a point $p \in \mathcal{P}$ and a set $A \subseteq \mathcal{P}$. The signed distance from p to A is defined as

$$Dist_d(p, A) := \begin{cases} -\inf \{d(p, p') \mid p' \in A\} & \text{if } p \notin A \\ \inf \{d(p, p') \mid p' \notin A\} & \text{if } p \in A \end{cases} \quad (5.10)$$

$d(p, p')$ denotes the distance from p to p' using some metric. In the following, the *Euclidean metric* $d(p, p') = \|p - p'\|$ is used.

The formal semantics of the STL robustness metric is stated next.

Definition 5.3. STL robustness semantics (Fainekos and Pappas, 2009):

The STL robustness $\llbracket \varphi \rrbracket_d(w, t_i)$ of a signal w w.r.t to the STL formula φ at time

instance t_i , $i \in [0, 1, \dots, N]$ is defined as:

$$\llbracket \top \rrbracket_d(w, t_i) := +\infty \quad (5.11a)$$

$$\llbracket \pi \rrbracket_d(w, t_i) := \text{Dist}_d(s_i, \mathcal{O}(\pi)) \quad (5.11b)$$

$$\llbracket \neg\varphi \rrbracket_d(w, t_i) := -\llbracket \varphi \rrbracket_d(w, t_i) \quad (5.11c)$$

$$\llbracket \varphi_1 \vee \varphi_2 \rrbracket_d(w, t_i) := \max(\llbracket \varphi_1 \rrbracket_d(w, t_i), \llbracket \varphi_2 \rrbracket_d(w, t_i)) \quad (5.11d)$$

$$\llbracket \bigcirc\varphi \rrbracket_d(w, t_i) := \begin{cases} \llbracket \varphi \rrbracket_d(w, t_{i+1}) & \text{if } i+1 \in N \\ -\infty & \text{otherwise} \end{cases} \quad (5.11e)$$

$$\llbracket \varphi_1 U_{\mathcal{I}} \varphi_2 \rrbracket_d(w, t_i) := \quad (5.11f)$$

$$\max_{j \text{ s.t. } (t_j - t_i) \in \mathcal{I}} \left(\min \left(\llbracket \varphi_2 \rrbracket_d(w, t_j), \min_{i \leq k < j} \llbracket \varphi_1 \rrbracket_d(w, t_k) \right) \right),$$

where s_i is the state at time t_i , π is a predicate, and $\mathcal{O}(\pi)$ is the corresponding set in \mathcal{S} . For short, $\llbracket \varphi \rrbracket(w)$ refers to the robustness of the signal w at time t_0 using the Euclidean norm as the metric.

5.4 Contract-based Design

Contract-based design is a design methodology that enables modular and independent design. Contract-based design divides a system into encapsulated design units called *components*. Each component is associated with an assume-guarantee contract which specifies what behavior the component assumes about the environment it operates in, and what behavior it can guarantee given that these assumptions hold. The ultimate goal of contract-based design is to enable compositional reasoning, that is, reasoning about the correctness of a system based on the contracts of individual components. Contract-based design can also structure and formalize the design process by enabling stepwise refinement. This means that a design process starts from a system-level component at a high level of abstraction, which is incrementally refined into more detailed sub-components (Sangiovanni-Vincentelli et al., 2012). Contract-based design first appeared in the context of software engineering, with the *design by contract* methodology of Meyer (1992). More recently, significant research efforts have been put into using contract-based design to address the design and verification challenges of complex cyber-physical systems (Sangiovanni-Vincentelli et al., 2012). Contract-based design has seen application in other industries which are faced with complex and safety-critical systems, such as aviation (Nuzzo et al., 2014) and automotive (Benveniste et al., 2008). However, in the maritime industry, there has not yet been a significant adoption, apart from the recent publication of Hake et al. (2021), which proposes a contract-based methodology for verifying software updates on shipboard equipment. In this thesis, **Paper B** applies contract-based design to the design and verification of autonomous vessels.

By using a suitable formal specification language for specifying contracts, it is possible to reason about the correctness of composition and refinement in a formal and mathematical manner (Cimatti and Tonetta, 2012). The mathematical foundation for **Paper B** is based on the assume-guarantee contract theory from Benveniste

et al. (2018). The main concepts of this theory are presented next. The reader is referred to Benveniste et al. (2018) for further details.

Recall the definition of behaviors in Section 5.2. A *component*, M , is described abstractly in terms of the set P of behaviors the component exhibits. The *composition* of two components $M_1 \times M_2$ is defined as the intersection of their respective sets of behaviors $P_1 \cap P_2$.

An assume-guarantee contract \mathcal{C} is defined as a pair of assertions (A, G) . Each of these assertions is also defined as a set of behaviors. A are called the assumptions and G are called the guarantees. The set $\mathcal{E}_{\mathcal{C}}$ of legal environments is the collection of all components E for \mathcal{C} , such that $E \subseteq A$. The set $\mathcal{M}_{\mathcal{C}}$ of legal implementations of \mathcal{C} is defined by the collection of components M such that $A \times M \subseteq G$. All contracts which admit the same set of behaviors are by definition equivalent. A contract is said to be *saturated* if $G = G \cup A^c$, where A^c is the complement of A . Saturation indicates that the set of guarantees is maximal in the sense that it contains all behaviors where the assumptions do not hold. All contracts can be transformed to an equivalent saturated contract by letting $G = G \cup A^c$.

Next, a set of contract operations are defined. Let $\mathcal{C}_1 = (A_1, G_1)$ and $\mathcal{C}_2 = (A_2, G_2)$ be two saturated contracts defined over the same set of variables. The refinement relation $\mathcal{C}_2 \leq \mathcal{C}_1$ is defined by

$$\mathcal{C}_2 \leq \mathcal{C}_1 \text{ iff } G_2 \subseteq G_1 \text{ and } A_2 \supseteq A_1. \quad (5.12)$$

If this relation is satisfied, \mathcal{C}_2 refines \mathcal{C}_1 . Refinement is an ordering of the relative strength of contracts. Informally, a contract has to have as strong or stronger guarantees and as permissive or more permissive assumptions as another contract to refine it.

The *conjunction* of \mathcal{C}_1 and \mathcal{C}_2 , denoted $\mathcal{C}_1 \wedge \mathcal{C}_2$, is defined as

$$\mathcal{C}_1 \wedge \mathcal{C}_2 := (A_1 \cup A_2, G_1 \cap G_2). \quad (5.13)$$

Conjunction can be used to impose several different contracts on a component, such that the component needs to comply with each of them in order to comply with the conjunction.

The *composition* of two contracts is denoted $\mathcal{C}_1 \otimes \mathcal{C}_2$. Composition is defined as

$$\mathcal{C}_1 \otimes \mathcal{C}_2 := (G_1 \cap G_2, (A_1 \cap A_2) \cup (G_1 \cap G_2)^c) \quad (5.14)$$

Composition of contracts can be used to construct composite contracts out of simpler ones. For instance, if a component is implemented by a set of sub-components, one may want to verify that the composition of the contracts for the sub-components refines the contract of the parent component. To comply with $\mathcal{C}_1 \otimes \mathcal{C}_2$, a component has to comply with the guarantees of both \mathcal{C}_1 and \mathcal{C}_2 . However, the assumptions of the composite contract are relaxed, as some of the assumptions may be covered

by the guarantees of the other contract. It can be shown that using these definitions, the conjunction and composition operators are associative and commutative.

Finally, the concept of *observers* for contracts is defined¹. Observers are in this context used for evaluating whether a single behavior complies with a contract or not, which is central when testing a component. For a contract $\mathcal{C} = (A, G)$ and a behavior σ , the observer $b_{\mathcal{C}}$ is defined by

$$b_{\mathcal{C}}(\sigma, A, G) = \top \text{ if and only if } \sigma \in G \cup A^c \quad (5.15)$$

5.5 Automated Theorem Proving

Automated theorem proving is a formal verification technique that uses automated reasoning by computer programs to prove mathematical theorems. Automated theorem proving tools can be divided into automatic theorem provers, such as Z3 (de Moura and Bjørner, 2008), and interactive theorem provers, such as Isabelle (Paulson, 1994) and Coq (Bertot, 2008), where the user interacts with a computer program to incrementally build a proof. Automatic theorem provers are generally easier to use and require less expertise from the user compared to interactive theorem provers. However, interactive theorem provers are generally more capable in terms of their theorem proving capabilities.

Automatic theorem provers are given a mathematical theorem as input and attempt to decide if it is true or false. The input theorems are specified using some formal logic, such as propositional logic. The Z3 automatic theorem is used in conjunction with contract-based design in **Paper B** in order to do formal and automatic compositional reasoning. Z3 is introduced in more detail in the following.

5.5.1 The Z3 Automatic Theorem Prover

Z3 is an open-source automatic theorem prover developed by Microsoft Research (de Moura and Bjørner, 2008). Z3 is implemented as a Satisfiability Modulo Theories (SMT) solver. To understand SMT solvers, the simpler and more well-known boolean satisfiability solvers (SAT solvers) are introduced first.

SAT solvers are computer tools that attempt to decide if a propositional logic formula is satisfiable or not. That is, given a propositional logic formula with Boolean variable p, q, r, \dots , does there exist a combination of true/false assignments to these variables which makes the propositional logic formula true? SAT solvers are the workhorse of a wide array of algorithms. Because the SAT problem is nondeterministic polynomial-time complete (NP-complete), a wide range of important NP-complete problems can be transformed into a SAT problem. Significant effort has therefore been put into developing efficient algorithms and heuristics for SAT solving. In fact, state-of-the-art SAT solvers can solve problem instances involving

¹Note that the term observer is used with a different meaning in **Paper C**. There, an observer refers to a state estimator.

several hundred thousand Boolean variables.

SMT solvers are a generalization of SAT solvers which can solve for a larger class of formulas. This is achieved by using more general logic formulas than propositional logic, such as first-order logic or higher-order logic, and by admitting more variable types than only boolean variables, such as integers, real numbers, arrays, or strings. SMT solvers combine a satisfiability solver with a set of domain-specific theories, such as linear arithmetic for integers, and real analysis for reals.

Z3 uses an SMT solver approach to theorem proving by checking the satisfiability of the negation of a theorem. If the negation of the theorem is found to be unsatisfiable, then the theorem is proved. If, on the other hand, a solution is found that satisfies the negation of the theorem, this will disprove the theorem, and the SMT will return the solution as a counterexample. Z3 theorems are specified as first-order logic formulas. In addition to the operators of propositional logic, first-order logic also has two *quantifiers*. The universal quantifier, *for all*, is denoted by the symbol \forall , and the existential quantifier, *there exists*, is denoted by the symbol \exists .

Z3 has bindings for several programming languages in addition to supporting the standardized SMT syntax SMT-LIB. Below, a simple example of using Z3 to prove a theorem in Python is given. Z3's *prove()* function returns a *proved* result when running this script.

```
x,y = Reals('x y')
triangle_inequality = Sqrt((x+y)**2) <= Sqrt(x**2) + Sqrt(y**2)
prove(triangle_inequality)
```

Chapter 6

Hybrid Systems

Hybrid systems exhibit both continuous and discrete behavior. Many phenomena can be modeled naturally as hybrid systems, a classical example is a bouncing ball whose states evolve continuously while the ball is in the air, but exhibits an instantaneous change of velocity when hitting the floor. Hybrid approaches also enable many new possibilities for control design (Goebel et al., 2012; Sanfelice, 2021) which have seen several applications for control of marine vessels (Brodtkorb, 2017). Prominent examples include a hybrid controller that enables dynamic positioning of underactuated vessels (Panagou and Kyriakopoulos, 2014), a hybrid dynamic positioning system that switches the control model, controller, and observer based on the environmental conditions (Nguyen et al., 2007), and a hybrid controller concept that enables provably stable switching between a set of candidate controllers (Brodtkorb et al., 2018). Hybrid systems theory also has applications for verification of cyber-physical systems. Software can be modeled by discrete finite-state systems, while the physical world can be modeled by continuous differential equations. Hybrid systems theory enables formal analysis of their combined behavior.

The first research on hybrid systems appeared in Witsenhausen (1966). Since this pioneering work, several different formalisms have been proposed for modeling hybrid systems. This chapter gives an introduction to a particular mathematical framework for the modeling and analysis of hybrid systems. This framework is used in the formulation and formal analysis of the resetting observer in **Paper C**.

6.1 Hybrid Dynamical Systems Framework

The framework used in **Paper C** is the Hybrid Dynamical Systems framework of Goebel et al. (2012). This framework extends well-known concepts from nonlinear systems theory to hybrid dynamical systems. The framework builds on some background theory from set-valued analysis which is beyond the scope to introduce here. The reader is referred to Rockafellar and Wets (1998) for further details on this.

A hybrid dynamical system, $\mathcal{H} = (C, F, D, G)$, is defined by a *constrained differ-*

ential inclusion and a constrained difference inclusion:

$$x \in C \quad \dot{x} \in F(x) \tag{6.1a}$$

$$x \in D \quad x^+ \in G(x) \tag{6.1b}$$

When x is in the set C , it evolves continuously (*flows*) according to the *set-valued mapping* $\dot{x} = f(x)$ for some $f \in F$. When x is in the set D , it evolves discretely (*jumps*) according to set-valued mapping $x^+ = g(x)$ for some $g \in G$. x^+ denotes the value of x after the jump.

Next, a precise definition of the solutions to a hybrid dynamical system $\mathcal{H} = (C, F, D, G)$ is given. To arrive at this, two other definitions are needed first. The first is the notion of a *hybrid time domain*.

Definition 6.1. Hybrid time domain (Goebel et al. (2012) Def. 2.3)

A subset $E \subset \mathbb{R}_{\geq 0} \times \mathbb{N}$ is a compact hybrid time domain if

$$E = \bigcup_{j=0}^{J-1} ([t_j, t_{j+1}], j)$$

for some finite sequence of times $0 = t_0 \leq t_1 \leq t_2 \dots \leq t_J$. It is a hybrid time domain if for all $(T, J) \in E$, $E \cap ([0, T] \times \{0, 1, \dots, J\})$ is a compact hybrid time domain.

Next, the notion of *hybrid arcs* is defined.

Definition 6.2. Hybrid arc (Goebel et al. (2012) Def. 2.4)

A function $\phi : E \mapsto \mathbb{R}^n$ is a hybrid arc if E is a hybrid time domain and if for each $j \in \mathbb{N}$, the function $t \mapsto \phi(t, j)$ is locally absolutely continuous on the interval $I^j := \{t : (t, j) \in E\}$.

A function is *locally absolutely continuous* if the derivative is continuous for almost all times, and the function can be recovered by integrating the derivative. Finally, the notion of a solution to $\mathcal{H} = (C, F, D, G)$ can be defined.

Definition 6.3. Solution to a hybrid system (Goebel et al. (2012) Def 2.6)

A hybrid arc ϕ is a solution to a hybrid system $\mathcal{H} = (C, F, D, G)$ if $\phi(0, 0) \in \bar{C} \cup D$, and

- (i) for all $j \in \mathbb{N}$ such that $I^j := \{t : (t, j) \in \text{dom}(\phi)\}$ has a nonempty interior

$$\phi(t, j) \in C \quad \text{for all } t \in \text{int}(I^j)$$

$$\dot{\phi}(t, j) \in F(\phi(t, j)) \quad \text{for almost all } t \in I^j$$

- (ii) for all $t, j \in \text{dom}(\phi)$ such that $(t, j + 1) \in \text{dom}(\phi)$

$$\phi(t, j) \in D$$

$$\phi(t, j + 1) \in G(\phi(t, j))$$

where \bar{C} is the closure of the set C , $\text{dom}(\phi)$ is the domain of ϕ and $\text{int}(I^j)$ is the interior of the set I^j .

6.2 Stability and Robustness

There exists an extensive toolbox of formal analysis results for hybrid dynamical, some of the most important results are presented next.

A special class of hybrid systems is *well-posed hybrid systems*, for which there exist several stability and robustness results. A system $\mathcal{H} = (C, F, D, G)$ is well-posed if it satisfies the *hybrid basic assumptions*:

Definition 6.4. Hybrid basic assumptions (Goebel et al. (2012), Assumption 6.5)

- (i) C and D are closed subsets of \mathbb{R}^n
- (ii) $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is outer semicontinuous and locally bounded relative to C , $C \subset \text{dom}(F)$, and $F(x)$ is a convex set for every $x \in C$.
- (iii) $G : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is outer semicontinuous and locally bounded relative to D , and $D \subset \text{dom}(G)$.

A set-valued mapping $M : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is *outer semicontinuous* (OSC) at x if for each sequence (x_i, y_i) with $y_i \in M(x_i) \quad \forall i$ which converges to (x, y) , then $y \in M(x)$. M is *locally bounded* (LB) if for each r there exists an R such that $M(r\mathbb{B}) \subset M(R\mathbb{B})$, where \mathbb{B} is a unit ball set. Finally, $M(x)$ is a *convex* set if each point on a line connecting two points in $M(x)$ is also in $M(x)$.

Next, the notion of stability of sets for a hybrid system is defined:

Definition 6.5. Uniform global pre-asymptotic stability (UGpAS) (Goebel et al. (2012) Def. 3.6)

For the hybrid system $\mathcal{H} = (C, F, D, G)$, the closed set \mathcal{A} is said to be

- uniformly globally stable if there exists a class- \mathcal{K}_∞ function α such that any solution ϕ to \mathcal{H} satisfies $|\phi(t, j)|_{\mathcal{A}} \leq \alpha(|\phi(0, 0)|_{\mathcal{A}})$ for all $(t, j) \in \text{dom}(\phi)$
- uniformly globally pre-attractive if for each $\epsilon > 0$ and $r > 0$ there exists a $T > 0$ such that, for any solution ϕ to \mathcal{H} with $|\phi(0, 0)|_{\mathcal{A}} \leq r$, $(t, j) \in \text{dom}(\phi)$ and $t + j \geq T$ imply $|\phi(t, j)|_{\mathcal{A}} \leq \epsilon$
- uniformly globally pre-asymptotically stable for \mathcal{H} if it is both uniformly globally stable and uniformly globally pre-attractive.

The distance from a point x to the set \mathcal{A} is defined by $|x|_{\mathcal{A}} := \inf_{y \in \mathcal{A}} |x - y|$. The term *pre-asymptotic* as opposed to asymptotic stability and *pre-attraction*, as opposed to attraction, allows maximal solutions that are not complete.

Lyapunov functions can be used to analyze the stability of sets for hybrid systems.

Definition 6.6. Lyapunov function candidate (Goebel et al. (2012) Def. 3.16)

A function $V : \text{dom}(V) \mapsto \mathbb{R}$ is said to be a Lyapunov function candidate for the hybrid system $\mathcal{H} = (C, F, D, G)$ if the following conditions hold:

1. $\bar{C} \cup D \cup G(D) \subset \text{dom}(V)$
2. V is continuously differentiable on an open set containing \bar{C}

where \bar{C} denotes the closure of C .

The following hybrid Lyapunov theorem can be used to establish UGpAS of a compact set.

Theorem 6.1 (Hybrid Lyapunov theorem (Goebel et al., 2009)). *Consider the hybrid system $\mathcal{H} = (C, F, D, G)$ satisfying the hybrid basic conditions, and the compact set $\mathcal{A} \subset \mathbb{R}^n$ satisfying $G(\mathcal{A} \cap D) \subset \mathcal{A}$. If there exists a Lyapunov function candidate V for $(\mathcal{H}, \mathcal{A})$ such that*

$$\langle \nabla V(x), f \rangle < 0 \quad \forall x \in C \setminus \mathcal{A}, f \in F(x) \quad (6.2a)$$

$$V(g) - V(x) < 0 \quad \forall x \in D \setminus \mathcal{A}, g \in G(x) \quad (6.2b)$$

then the set \mathcal{A} is pre-asymptotically stable and the basin of pre-attraction contains every forward invariant compact set.

For systems satisfying the hybrid basic conditions, GpAS is equivalent to UGpAS. For these systems, GpAS also implies robust GpAS. This ensures that vanishing perturbations do not dramatically change the behavior of solutions.

Next is stated a relaxed version of Theorem 6.1, which gives sufficient conditions for UGpAS of a non-compact set in the case where there is non-decrease during jumps, strict decrease during flow, and the duration of flow is sufficiently large. This theorem is used to prove UGpAS of the resetting observer in **Paper C**.

Theorem 6.2 (Sufficient Lyapunov Conditions; Persistent Flowing (Goebel et al., 2012)). *Let $\mathcal{H} = (C, F, D, G)$ be a hybrid system, and let $\mathcal{A} \subset \mathbb{R}^n$ be closed. Suppose V is a Lyapunov function candidate for \mathcal{H} and there exist $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and a continuous $\rho \in \mathcal{PD}$ such that*

$$\alpha_1(\|x\|_{\mathcal{A}}) \leq V(x) \leq \alpha_2(\|x\|_{\mathcal{A}}) \quad \forall x \in C \cup D \cup G(D) \quad (6.3a)$$

$$\langle \nabla V(x), f \rangle \leq -\rho(\|x\|_{\mathcal{A}}) \quad \forall x \in C, f \in F(x) \quad (6.3b)$$

$$V(g) - V(x) \leq 0 \quad \forall x \in D, g \in G(x) \quad (6.3c)$$

If, for each $r > 0$, there exists $\gamma_r \in \mathcal{K}_\infty$, $N_r > 0$ such that for every solution ϕ to \mathcal{H} , $\|\phi(0, 0)\|_{\mathcal{A}} \in (0, r]$, $(t, j) \in \text{dom}(\phi)$, $t + j \geq T$ imply $t \geq \gamma_r - N_r$, then \mathcal{A} is uniformly globally pre-asymptotically stable for \mathcal{H} .

Chapter 7

Discussion of Results

The main research objective of this thesis was to investigate how formal approaches to the design and verification of control systems could contribute to solving the safety assurance challenges for autonomous vessels. An additional research objective was to improve the control system performance of autonomous vessels through innovations at the GNC level. Three research questions were formulated to address the research objectives. This chapter answers the research questions by summarizing and discussing the findings from the research papers.

7.1 Research Question 1

RQ1: What are the main challenges for safety assurance of control systems for autonomous vessels?

The following summarizes the identified safety assurance challenges that have been aggregated through the doctoral work. The challenges are grouped into three main categories:

- Autonomy-specific challenges
- Challenges related to complexity and software
- Maritime-specific challenges

The identified challenges are detailed in the following.

7.1.1 Autonomy-specific Challenges

The root cause of the identified safety assurance challenges for autonomy arises from the extensive sensing and interaction with the open environment (**Papers A, B and D**). This is well known from automotive autonomy (Koopman and Wagner, 2017) and has also been identified for maritime autonomy in e.g. (Murray et al., 2022). This characteristic effectively means that an autonomous system may encounter infinitely many scenarios during operation, and it will thus never be possible to specify the required behavior in every scenario during design. This challenge

is novel for autonomy because traditional automation systems have been limited to well-defined tasks and interactions with the environment, while the human operator has been responsible for sensing and interacting with the open environment. Human operators are extremely skilled at resolving new situations, and transferring this skill set to machines is a challenge both for design and safety assurance.

One attempt at addressing this challenge is the use of machine learning algorithms that learn from data instead of being programmed from a specification (**Paper D**). For autonomous vessels, this is particularly relevant for the camera-based object detection algorithms. While such algorithms have been quite successful at solving unspecifiable problems, they introduce an even greater challenge for safety assurance due to lack of explainability and being sensitive to gaps or unintended biases in the training data (Rao and Frtunikj, 2018). Major autonomy players in the automotive industry have attempted to use machine learning with massive training data collection through large vehicle fleets and large-scale simulations. However, they have still not been able to achieve the necessary reliability due to the long-tailed distribution of edge cases.

Another novelty for autonomous systems is the use of high-dimensional perception sensors to sense the environment and gain situational awareness (**Paper D**). Traditional automation systems have mainly used sensors that collect scalar data, such as position, pressure, or temperature. In contrast, perception sensors collect high-dimensional data, such as camera images, lidar point clouds, and radar scans. This drastically increases the input space and limits the feasibility of mathematical reasoning about properties such as correctness, stability, and robustness of the algorithms that process the high-dimensional data. Simulation of such sensors is possible using 3D rendering technology (Vasstein et al., 2020), but the simulation models are computationally expensive, and validating them against real behavior is an open research problem.

7.1.2 Challenges Related to Complexity and Software

The next class of identified challenges is due to autonomous vessels being highly complex and software-intensive (**Papers B and D**). Safety assurance of complex systems is challenging because safety is a property that emerges from all the interactions between the constituents of the system (Haugen, 2019). Identifying, analyzing, and understanding all the interactions in a complex system is almost impossible to do a priori. Autonomous vessel control systems are highly software intensive and introduce both increased complexity and criticality. Software can introduce intricate and hard-to-predict failure mechanisms (Thieme et al., 2020). Moreover, state-of-the-art development processes for safety-critical software are not well adapted to the complexity and types of algorithms used in autonomous vessel control systems. Another challenge with software-intensive systems is life-cycle change management. Software is typically subject to updates and finding efficient processes for ensuring that a system continues to be safe after updates is an identified challenge (Hake et al., 2021).

7.1.3 Maritime-specific Challenges

The challenges identified thus far are generally valid for most autonomous systems and are well-known in the automotive industry. Next, some identified challenges which are specific to maritime autonomy are discussed.

Precise motion control is generally harder for autonomous vessels than cars (**Papers C and G**). When the motion planner of a car provides a motion reference, it can be certain the car will track this reference with high precision because the motion of the car is precisely determined by the speed and the steering angle. Maritime vessels, on the other hand, glide through the water under the influence of wind, waves, and currents. Hence, there may be significant discrepancies between the planned motion and the actual motion, which can be critical in situations with small margins, such as docking or passing through narrow straits (**Paper F**). This complicates motion planning, as it must consider these uncertainties.

Another challenge for maritime autonomy is that autonomous vessel internal systems generally have more complexity than automotive systems (**Paper F**). The most significant difference is the presence of large power and propulsion systems, which require significant monitoring and control by the autonomous vessel control systems. The power and propulsion systems introduce a lot of potential failure modes which must be reliably detected and handled by the autonomy system (Johansen et al., 2007). These challenges are, however, less prominent for smaller vessels with fully electric propulsion.

A final identified challenge for maritime autonomy is the limited access to operational experience for autonomous vessels (**Paper D**). The automotive industry has deployed millions of vehicles driving autonomously under human supervision and continuously collecting data and building experience. As the number of autonomous vessels will be far lower than the number of autonomous cars, combined with the tendency to use customized components and do one-of-a-kind builds, the maritime industry cannot rely on the same level of operational experience before deploying autonomous vessels. This puts even more emphasis on the need for rigorous design and verification processes for autonomous vessels.

On a positive note, there are several characteristics of maritime autonomy that may make it easier to realize than automotive autonomy. Generally, there are larger safety margins, lower speed, and more room for navigation. Moreover, many vessels travel along a fixed route, which limits the amount unknown scenarios compared to general autonomy, which needs to function everywhere. Another simplifying factor of maritime autonomy is that autonomous vessels don't need to be fully autonomous or unmanned to provide value. For instance, many vessels have multiple crew members, and providing autonomy functions that reduce the workload of navigators and engineers may enable crew reduction. This can be a stepping stone towards higher degrees of autonomy.

7.2 Research Question 2

RQ2: How can formal approaches for design and verification contribute to solving the identified challenges from RQ1

The research of this thesis has identified two key needs for addressing the challenges from RQ1:

1. **Increased formality:** Due to the complexity, criticality, huge scenario space, and limited access to operational experience, increased formality in the design and verification processes will be necessary for sufficiently addressing the safety assurance challenges. Increased formality can reduce the occurrence of defects, such as software bugs and design flaws, in the control system. This has been proven in practice in critical systems in other industries (Gerhart et al., 1994). Moreover, increased formality may be key to managing the complexity, for instance by designing components with inherent safety guarantees and integrating them in a formal way, thereby constraining the possible system interactions and outcomes (Sangiovanni-Vincentelli et al., 2012).
2. **Extensive simulation-based testing:** Due to the complexity, high-dimensional perception sensors, and advanced control algorithms of autonomous vessel control systems, simulation is the only viable and scalable way to assess system-level behavior. This conclusion is consistent with other research on simulation-based testing for autonomous vessels (Pedersen et al., 2020) and the approach taken by major actors for automotive autonomy (Aptiv et al., 2019; Huang et al., 2016). Moreover, due to the infinite amount of possible scenarios, simulation-based testing at a massive scale will be necessary to gain sufficient test coverage.

The research papers of this thesis have both developed novel methodologies and reviewed the literature for existing methodologies that support increased formality and simulation-based testing. Specific contributions from the research papers are detailed in the following.

Paper A proposed the use of the formal logic STL in conjunction with a statistical Gaussian process (GP) model to formalize and automate simulation-based testing. Formalization is achieved both by formal specification of requirements and by enabling quantification of the confidence level and test coverage. Automation is achieved because STL enables automatic evaluation of simulations against requirements and because the Gaussian process model enables automatic and adaptive test case selection. Automation is key to enabling simulation-based at a massive scale. An additional contribution of **Paper A** is the formalization of a subset of COLREG using STL, which was developed in the case study of the paper. The methodology of **Paper A** also provided value as a design tool, because the STL robustness surface plot gives a map of the decision boundaries of the control system. In the planning algorithms and high-level decision-making of autonomous vessel control systems, the decision boundaries are typically implicitly specified as part of cost functions in the optimization solvers, and therefore hard to assess á

priori.

Paper B proposed the use of contract-based methods for the design and verification of autonomous vessel control systems. The methodology of **Paper B** gives increased formality by defining a structured design process and providing a formal syntax for specifying assume-guarantee contracts. The contracts make the valid environments that components can operate in explicit through the assumptions, and constrains the possible behaviors through the guarantees. Moreover, it enables formal verification of contract refinement and compatibility by using the Z3 automated theorem prover. **Paper B** also proposed how to incorporate simulation-based testing in a contract-based setting, in a mutually beneficial way. Contract-based design is also a modular approach to design and verification, and modularity is a well-known strategy for managing system complexity. **Paper B** also proposed how modular contract-based verification can enable efficient re-verification to address the change management challenge for software-intensive systems. Finally, **Paper B** proposed monitoring of assumptions as an approach to address the key autonomy challenge of infinitely many unknown scenarios. The rationale behind this approach is that the control system does not need to be capable of solving any scenario to be safe, it only needs to know when it is in a scenario that it is not designed to handle. Knowing this, the system can, for instance, enter a minimum risk condition (MRC). Online monitoring of the contract assumptions may be an effective approach to detect when the system is in a situation it is not designed to handle.

Paper C proposed a resetting observer for linear time-varying systems which has applications for DP. While the observer itself mostly addresses RQ3, the formulation and verification of the observer are relevant for RQ2. The observer was formulated as a hybrid dynamical system using the framework of Chapter 6. Formulating the design in a formal framework enabled formal analysis of the stability of the observer, which is a great asset when including a quite complex nonlinear component with an intricate resetting logic in the control loop.

Paper D addresses RQ2 by reviewing the literature for existing FMs that have the potential to address the safety assurance challenges for autonomous vessel control systems by increasing the formality in the design and verification process. The paper identifies the use of formal specification languages as a promising tool for producing clear, unambiguous, and consistent requirements, which is a prerequisite for a good design. Moreover, formal specification of a design enables mathematical analysis, such as refinement checking and consistency checking. Such analysis can be automated by computers since formal specification languages are machine-readable. Next, **Paper D** identifies formal methodologies for transforming a formal specification into an implementation. Automatic code generation from a formal specification and synthesis of correct-by-construction controllers were highlighted as interesting methodologies. The paper also notes that manual programming from a formal specification can be instrumental in producing a good implementation, because it is easier to produce correct code from a clear specification, and because many fundamental flaws are eliminated at the design stage. Finally, **Paper**

D reviewed formal verification techniques. Model checking was highlighted as a promising candidate, which may have particular applications for verifying high-level decision-making and mode control for autonomous vessels, as these modules often are implemented as finite-transition systems. Automated theorem proving was identified as a promising methodology to prove the correctness of control algorithms. To give a balanced presentation of FMs, **Paper D** also discusses some key limitations, such as limited scalability, the reality gap problem, and the fact that FMs are perceived as difficult and time-consuming to apply.

Paper E is a follow-up on suggested future work from **Paper A**, by integrating the automatic testing methodology using STL and GP with hazard identification from STPA. STPA is identified as a promising candidate for increasing the completeness of safety requirements in complex systems. In **Paper E**, STPA was used to identify STL test requirements and parametric test cases for simulation-based testing using the methodology of **Paper A**.

Paper F proposed a methodology for online risk management. STPA was used for hazard identification, which forms the basis for an online risk model in form of a Bayesian belief network (BBN). The online risk model gives input to an optimization solver which balances risk with other operational aspects in order to achieve risk-aware behavior. The paper used the automatic simulation-based testing methodology of **Paper A** to verify the online risk management system against formal requirements in STL.

7.3 Research Question 3

RQ3: How can innovations at the GNC level contribute to improved control system performance for autonomous vessels?

As Figure 2.3 shows, autonomous functionality is built on top of a foundation of automation systems, of which the GNC functionality is a significant part. The performance, robustness, and safety of an autonomous vessel control system are therefore highly dependent on good GNC functionality (DNV, 2018). Moreover, RQ1 highlighted the uncertainty in motion control as an important challenge for maritime autonomy, as it complicates motion planning. Increasing the performance of the GNC systems is key to reducing the uncertainty in motion control. Improvements to the GNC systems are addressed by **Paper C** and **Paper G**.

Paper C proposed a hybrid observer which was shown to significantly increase the reactivity to unmodeled disturbances and modeling errors compared to the state-of-the-art DP observer. Observers produce the state estimates which provide the feedback signal to the DP controller in the motion control loop. Inaccuracies and phase-lags in the state estimates from the observer can therefore have a severely detrimental effect on the motion control performance. DP systems have traditionally been designed for station keeping, which prioritizes steady-state performance and noise filtering over reactivity. However, using DP systems for motion control

of autonomous vessels have much higher requirements to reactivity. Transitioning from high-speed transit to low-speed docking, and achieving sufficiently fast docking are well-known challenges for DP systems used in autonomous vessels. The observer of **Paper C** can contribute to addressing this.

Paper G proposed a novel control allocation algorithm for double-ended ferries. Double-ended ferry operations are particularly interesting for piloting maritime autonomy due to their relatively low mission complexity. Improper control allocation can have a detrimental effect on the motion control performance, by introducing delays and disturbances in the control loop. The control allocation algorithms of **Paper G** appeared to outperform state-of-the-art control allocation algorithms by taking advantage of the symmetry in the thruster configuration of double-ended ferries which enabled fast and robust solving of the fully nonlinear allocation problem.

Chapter 8

Conclusions and Future Work

This final chapter concludes the thesis and provides some suggestions for future work.

8.1 Conclusions

The main research objective of this thesis was to investigate how formal approaches to the design and verification of control systems can contribute to solving the safety assurance challenges for autonomous vessel control systems. To thesis first investigated the main challenges, and grouped the identified challenges into three categories: autonomy-specific challenges, challenges related to complexity and software, and maritime-specific challenges. Next, the thesis investigated how formal approaches for the design and verification of autonomous vessel control systems can contribute to addressing these challenges. Two key needs for addressing the challenges were identified: Increased formality to manage the complexity and criticality, and extensive use of simulation-based testing to be able to scalably assess the complex system-level behavior over the huge scenario space.

The research papers of the thesis developed novel methodologies to address these needs. Some key contributions were:

- A methodology which used the formal logic STL in conjunction with a GP model to formalize and automate simulation-based testing. This enabled simulation-based testing at a massive scale, with quantification of the test space coverage and confidence level.
- A methodology for contract-based design and verification that combined simulation-based testing with formal compositional reasoning using an automated theorem prover.
- The use of hybrid systems theory for formulation and formal stability analysis of a novel resetting observer design.

In addition to developing new methodologies, the thesis reviewed the literature on FMs and identified several existing tools and methodologies which have the potential to address the safety assurance challenges for autonomous vessel control

systems. Key identified applications included formal specification, code generation, correct-by-construction controller synthesis, model checking of supervisory control logic, and automated theorem proving of planning algorithms.

An additional research objective of the thesis was to improve the control system performance of autonomous vessels through innovations at the GNC level. GNC functionality forms the foundation of autonomous vessel control systems and is vital for safe and robust autonomy. Moreover, uncertainties in the motion control were identified as a key challenge for safety assurance of maritime autonomy, and improvement of GNC functionality is key to reducing this uncertainty. The thesis contributed to this research objective by developing a more reactive observer design, and by developing a novel control allocation algorithm for double-ended ferries.

In conclusion, formal approaches for design and verification appear as a promising direction for addressing some of the safety assurance challenges for autonomous vessel control systems, and this thesis has developed several specific examples of how this can be done. Formal approaches are, however, not a universal solution, and the thesis has also identified several well-known limitations, such as limited scalability and the reality gap problem. Hence, it is important to see formal approaches in conjunction with informal approaches. More broadly, a hope for this thesis was to introduce FMs to the maritime audience and trigger more research and development in this direction and ultimately contribute to the realization of safe autonomous vessels.

8.2 Future Work

The use of FMs for autonomous vessel control systems is largely an unexplored field of research, and this thesis has barely scratched the surface. Moreover, several specific promising applications of FMs were identified but not pursued by this thesis, and thus remain for future work. Some particularly interesting directions for future work are:

- Online verification of planned behavior, using a digital twin for predicting future behavior in a set of relevant scenarios. This can for instance be achieved using the methodology of **Paper A** in an online setting.
- Online monitoring (also known as runtime monitoring). In particular, **Paper B** proposed online monitoring of assumptions in the assume-guarantee contracts as a key direction for future work.
- Model checking of supervisory control. Supervisory control is often implemented as a finite-state machine that handles critical discrete events such as operator inputs, operational mode management, failure response, and MRC selection. This can be modeled as a set of concurrent processes that interact asynchronously, which appears to be a good fit for model checking. This can for instance verify the absence of deadlocks or data races.
- Automated theorem proving for verifying the correctness of control and planning algorithms, such as collision avoidance protocols.

Additionally, each of the research papers has proposed directions for future work for their respective methodologies.

References

- Alur, R. and Henzinger, T.A. (1993). Real-Time Logics: Complexity and Expressiveness. *Information and Computation*, 104, 35–77.
- Aptiv, Audi, Baidu, BMW, Continental, Daimler, FCA, HERE, Infineon, Intel, and Volkswagen (2019). Safety first for automated driving. Technical report.
- Benveniste, A., Caillaud, B., Ferrari, A., Mangeruca, L., Passerone, R., and Sofronis, C. (2008). Multiple Viewpoint Contract-Based Specification and Design. In de Boer, F. S., Bonsangue, M. M., Graf, S., and de Roever, W.-P. (eds.), *Formal Methods for Components and Objects*, 200–225. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Raclet, J.B., Reinkemeier, P., Sangiovanni-Vincentelli, A., Damm, W., Henzinger, T.A., and Larsen, K.G. (2018). *Contracts for System Design*, volume 12. Now.
- Bertot, Y. (2008). A short presentation of Coq. In Mohamed, O. A., Muñoz, C., and Tahar, S. (eds.), *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5170 LNCS, 12–16. Springer Berlin Heidelberg, Berlin, Heidelberg. doi:10.1007/978-3-540-71067-7{-}3.
- Blanke, M., Frei, C.W., Kraus, F., Pat, R.J., and Staroswiecki, M. (2000). What Is Fault-Tolerant Control? *IFAC Proceedings Volumes*, 33(11), 41–52. doi:10.1016/S1474-6670(17)37338-X. URL https://ac.els-cdn.com/S147466701737338X/1-s2.0-S147466701737338X-main.pdf?_tid=33103032-0a58-44a3-a226-0fa34b9e775a&acdnat=1535801381_7b312f3e03a22ee315c649550486085d.
- Brodtkorb, A.H. (2017). *Hybrid Control of Marine Vessels*. Ph.D. thesis, Norwegian University of Science and Technology.
- Brodtkorb, A.H., Værnø, S.A., Teel, A.R., Sørensen, A.J., and Skjetne, R. (2018). Hybrid controller concept for dynamic positioning of marine vessels with experimental results. *Automatica*, 93, 489–497. doi:10.1016/j.automatica.2018.03.047. URL <https://doi.org/10.1016/j.automatica.2018.03.047>.
- CBC (2022). B.C. Ferries cancels at least a dozen sailings, blames lack of staff. URL <https://www.cbc.ca/news/canada/british-columbia/bc-ferries-cancellation-aug-20-1.6557590>.

- Chen, C.T. (1999). *Linear System Theory and Design*. Oxford University Press, third edition. doi:10.1016/0005-1098(86)90039-7.
- Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., and Tacchella, A. (2002). NuSMV 2: An opensource tool for symbolic model checking. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2404, 359–364.
- Cimatti, A. and Tonetta, S. (2012). A property-based proof system for contract-based design. *Proceedings - 38th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2012*, 21–28. doi:10.1109/SEAA.2012.68.
- de Moura, L. and Bjørner, N. (2008). Z3: An Efficient SMT Solver. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 337–340. Springer Berlin Heidelberg, Berlin, Heidelberg.
- DNV (2018). Remote-Controlled and Autonomous Ships. Technical report, DNV. URL https://maritimecyprus.files.wordpress.com/2018/09/dnv_gl_autonomous_ships_2018-08.pdf.
- Fainekos, G.E. and Pappas, G.J. (2009). Robustness of temporal logic specifications for continuous-time signals. *Theoretical Computer Science*, 410(42), 4262–4291.
- Fossen, T.I., Breivik, M., and Skjetne, R. (2003). Line-of-sight path following of underactuated marine craft. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 36(21), 211–216. doi:10.1016/S1474-6670(17)37809-6.
- Fossen, T.I., Strand, J.P., Fossen, T. I. & Strand, J.P., Fossen, T.I., Strand, J.P., and Fossen, T. I. & Strand, J.P. (1999). Passive nonlinear observer design for ships using Lyapunov methods. *Automatica*, 35(1), 3–16.
- Foster, S., Gleirscher, M., and Calinescu, R. (2020). Towards Deductive Verification of Control Algorithms for Autonomous Marine Vehicles. In *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems*, 113–118.
- Gerhart, S., Craigen, D., and Ralston, T. (1994). Experience with Formal Methods in Critical Systems. *IEEE Software*, 11(1), 21–28. doi:10.1109/52.251198.
- Goebel, R., Sanfelice, R.G., and Teel, A.R. (2009). Hybrid Dynamical Systems. *IEEE Control Systems Magazine*, 29(2). doi:10.1016/j.jneuroim.2013.05.008.
- Goebel, R., Sanfelice, R.G., and Teel, A.R. (2012). *Hybrid Dynamical Systems : Modeling, Stability, and Robustness*. Princeton University Press. URL https://books.google.no/books?hl=en&lr=&id=Jwr0Y03fuGQC&oi=fnd&pg=PP1&dq=info:Tl8gDI003QEJ:scholar.google.com&ots=TpU8ELWhbC&sig=r7zQ6kma2MhE_uw-qkAhnS2oQNI&redir_esc=y#v=onepage&q&f=false.

- Hake, G., Hohl, C.P., and Hahn, A. (2021). Continuous Contract Based Verification of Updates in Maritime Shipboard Equipment. *J. Mar. Sci. Eng*, 9(7). doi:10.3390/jmse9070688. URL <https://doi.org/10.3390/jmse9070688>.
- Haugen, O.I. (2019). Safety assurance of complex systems. Technical report, DNV.
- Holzmann, G. (1997). The model checker SPIN. *IEEE Transactions on Software Engineering*, 23(5), 279–295. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=588521>.
- Huang, W.L., Wang, K., Lv, Y., and Zhu, F.H. (2016). Autonomous vehicles testing methods review. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 163–168. doi:10.1109/ITSC.2016.7795548.
- Huth, M. and Mark, R. (2004). *Logic in Computer Science*. Cambridge University Press, second edition.
- IEC (2010). IEC 61508 Functional safety of electrical/electronic/programmable electronic safety-related systems.
- ISO (2018). ISO26262 Road vehicles — Functional safety.
- ISO/PAS (2022). ISO/PAS 21448 Road vehicles - Safety of the intended functionality.
- Johansen, T. and Utne, I.B. (2022). Supervisory risk control of autonomous surface ships. *Ocean Engineering*, 251(March). doi:10.1016/j.oceaneng.2022.111045.
- Johansen, T.A. and Fossen, T.I. (2013). Control allocation - A survey. *Automatica*, 49(5), 1087–1103. doi:10.1016/j.automatica.2013.01.035.
- Johansen, T.A., Sørensen, A.J., Nordahl, O.J., Mo, O., and Fossen, T.I. (2007). Experiences from Hardware-in-the-loop (HIL) Testing of Dynamic Positioning and Power Management Systems. In *OSV Singapore*, 41–50.
- Kapinski, J., Deshmukh, J.V., Jin, X., Ito, H., and Butts, K. (2016). Simulation-Based Approaches for Verification of Embedded Systems. *IEEE Control Systems Magazine*, 36(November).
- Khalil, H.K. (2002). *Nonlinear Systems*. Prentice-Hall, 2 edition. doi:10.1007/978-1-4757-3108-8.
- Koopman, P., Ferrell, U., Fratrick, F., and Wagner, M. (2019). A Safety Standard Approach for Fully Autonomous Vehicles. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11699 LNCS(19), 326–332. doi:10.1007/978-3-030-26250-1_{_}26.
- Koopman, P. and Wagner, M. (2017). Autonomous Vehicle Safety: An Interdisciplinary Challenge. *IEEE Intelligent Transportation Systems Magazine*, 9(1), 90–96. doi:10.1109/MITS.2016.2583491.

- Krasowski, H. and Althoff, M. (2021). Temporal Logic Formalization of Marine Traffic Rules. In *IEEE Intelligent Vehicles Symposium (IV)*, 186–192.
- Larsen, K.G., Petterson, P., and Yi, W. (1997). UPPAAL in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1, 134–152.
- Lecomte, T., Servat, T., and Pouzancre, G. (1991). Formal Methods in Safety Critical Systems. *Safety and Reliability*, 11(4), 6–17.
- Leveson, N. (2011). *Engineering A Safer World*. The MIT Press, Cambridge. doi:10.1080/13623699.2017.1382166.
- Leveson, N. and Thomas, J. (2018). *STPA Handbook*. URL <http://files/625/Leveson-Thishandbookisintendedforthoseinterestedin.pdf>.
- Lu, Y., Niu, H., Savvaris, A., and Tsourdos, A. (2016). Verifying Collision Avoidance Behaviours for Unmanned Surface Vehicles using Probabilistic Model Checking. *IFAC-PapersOnLine*, 49(23), 127–132. doi:10.1016/j.ifacol.2016.10.332.
- Luckcuck, M., Farrell, M., Dennis, L.A., Dixon, C., and Fisher, M. (2019). Formal specification and verification of autonomous robotic systems: A survey. *ACM Computing Surveys*, 52(5), 100. doi:10.1145/3342355.
- Maler, O. and Nickovic, D. (2004). Monitoring temporal properties of continuous signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, 152–166. Springer.
- Martelli, M. and Figari, M. (2022). A design framework for combined marine propulsion control systems: From conceptualisation to sea trials validation. *Ocean Engineering*, 254(April), 111282. doi:10.1016/j.oceaneng.2022.111282. URL <https://doi.org/10.1016/j.oceaneng.2022.111282>.
- Meyer, B. (1992). Applying 'design by contract'. *Computer*, 25(10), 40–51.
- Meyer, P.J., Davonport, A., and Arcak, M. (2021). *Interval Reachability Analysis*. Springer Cham. doi:<https://doi.org/10.1007/978-3-030-65110-7>.
- Meyer, P.J., Yin, H., Brodtkorb, A.H., Arcak, M., and Sørensen, A.J. (2020). Continuous and discrete abstractions for planning, applied to ship docking. *IFAC-PapersOnLine*, 53(2), 1831–1836. doi:10.1016/j.ifacol.2020.12.2345. URL <https://doi.org/10.1016/j.ifacol.2020.12.2345>.
- Murray, B., Rødseth, O.J., Nordahl, H., Wennersberg, L.A.L., Pobitzer, A., and Foss, H. (2022). Approvable AI for Autonomous Ships: Challenges and Possible Solutions. In *Proceedings of the 32nd European Safety and Reliability Conference (ESREL 2022)*, 1975–1982. doi:10.3850/978-981-18-5183-4.
- Nguyen, T.D., Sørensen, A.J., and Tong Quek, S. (2007). Design of hybrid controller for dynamic positioning from calm to extreme sea conditions. *Automatica*, 43(5), 768–785. doi:10.1016/j.automatica.2006.11.017.

- NRK (2022). Nord-Norges mest trafikkerte fergesamband har innstilt 82 avganger i juli. URL https://www.nrk.no/nordland/torghatten-har-innstilt-82-avganger-mellom-bognes--lodingen--_blir-ikke-bedre-med-det-forste-1.16049456.
- Nuzzo, P., Sangiovanni-Vincentelli, A.L., Bresolin, D., Geretti, L., and Villa, T. (2015). A Platform-Based Design Methodology With Contracts and Related Tools for the Design of Cyber-Physical Systems. *Proceedings of the IEEE*, 103(11), 2104–2132. doi:10.1109/JPROC.2015.2453253.
- Nuzzo, P., Xu, H., Ozay, N., Finn, J.B., Sangiovanni-Vincentelli, A.L., Murray, R.M., Donzé, A., and Seshia, S.A. (2014). A contract-based methodology for aircraft electric power system design. *IEEE Access*, 2, 1–25. doi:10.1109/ACCESS.2013.2295764.
- Panagou, D. and Kyriakopoulos, K.J. (2014). Dynamic positioning for an under-actuated marine vehicle using hybrid control. *International Journal of Control*, 87(2), 264–280. doi:10.1080/00207179.2013.828853. URL <https://doi.org/10.1080/00207179.2013.828853>.
- Park, J. and Kim, J. (2020). Autonomous docking of an unmanned surface vehicle based on reachability analysis. In *International Conference on Control, Automation and Systems*, 962–966.
- Paulson, L.C. (1994). *Isabelle: A generic theorem prover*, volume 828. Springer Science & Business Media.
- Pedersen, T.A., Glomsrud, J.A., and Haugen, O.I. (2020). Towards Simulation-based Verification of Autonomous Navigation Systems. *Proceedings of the International Seminar on Safety and Security of Autonomous Vessels (ISSAV) and European STAMP Workshop and Conference (ESWC) 2019*, (December), 1–13. doi:10.2478/9788395669606-001.
- Pnueli, A. (1977). The temporal logic of programs. In *Proceedings - Annual IEEE Symposium on Foundations of Computer Science*, 1977, 46–57. IEEE. doi:10.1109/sfcs.1977.32.
- Porathe, T., Hoem, A., Rødseth, O., Fjørtoft, K., and Johnsen, S.O. (2018). At least as safe as manned shipping? Autonomous shipping, safety and “human error”. *Safety and Reliability - Safe Societies in a Changing World - Proceedings of the 28th International European Safety and Reliability Conference, ESREL 2018*, 417–426.
- Radan, D. (2008). *Integrated Control of Marine Electrical Power Systems*. URL https://pdfs.semanticscholar.org/b770/71c49cf7425e23a386e5d90b779cd21e242e.pdf%0Ahttp://folk.ntnu.no/assor/PhDThesis/Phd_Radan_NTNU.pdf.
- Rao, Q. and Frtunikj, J. (2018). Deep learning for self-driving cars: Chances and challenges: Extended Abstract. *Proceedings - International Conference on Software Engineering*, 35–38. doi:10.1145/3194085.3194087.

- Rockafellar, R.T. and Wets, R.J.B. (1998). *Variational Analysis*. Springer Berlin, Heidelberg, 1 edition.
- Rødseth, O.J., Wennersberg, L.A.L., and Nordahl, H. (2022). Levels of autonomy for ships. *Journal of Physics: Conference Series*, 2311(1). doi:10.1088/1742-6596/2311/1/012018.
- Rokseth, B. (2018). *Safety and Verification of Advanced Maritime Vessels: An Approach Based on Systems Theory*. Ph.D. thesis, Norwegian University of Science and Technology.
- Rokseth, B., Haugen, O.I., and Utne, I.B. (2019). Safety Verification for Autonomous Ships. *MATEC Web of Conferences*, 273, 02002. doi:10.1051/mateconf/201927302002.
- Sanfelice, R.G. (2021). *Hybrid Feedback Control*. Princeton University Press.
- Sangiovanni-Vincentelli, A., Damm, W., and Passerone, R. (2012). Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. *European Journal of Control*, 18(3), 217–238. doi:10.3166/EJC.18.217-238. URL <http://dx.doi.org/10.3166/ejc.18.217-238>.
- Shokri-Manninen, F., Vain, J., and Waldén, M. (2020). Formal Verification of COLREG-Based Navigation of Maritime Autonomous Systems. In *Lecture Notes in Computer Science, Software Engineering and Formal Methods*, 41–59.
- Sørensen, A.J. (2011). A survey of dynamic positioning control systems. *Annual Reviews in Control*, 35(1), 123–136.
- Sørensen, A.J. and Ludvigsen, M. (2018). Underwater Technology Platforms. *Encyclopedia of Maritime and Offshore Engineering*, 1–11. doi:10.1002/9781118476406.emoe323.
- Tabuada, P. (2009). *Verification and Control of Hybrid Systems*. Springer.
- Thieme, C.A., Mosleh, A., Utne, I.B., and Hegde, J. (2020). Incorporating software failure in risk analysis – Part 1: Software functional failure mode classification. *Reliability Engineering and System Safety*, 197(August 2019), 106803. doi:10.1016/j.ress.2020.106803. URL <https://doi.org/10.1016/j.ress.2020.106803>.
- Thombre, S., Zhao, Z., Ramm-Schmidt, H., Vallet Garcia, J.M., Malkamaki, T., Nikolskiy, S., Hammarberg, T., Nuortie, H., Bhuiyan, M.Z.H., Särkkä, S., and Lehtola, V.V. (2022). Sensors and AI Techniques for Situational Awareness in Autonomous Ships: A Review. *IEEE Transactions on Intelligent Transportation Systems*, 23(1), 64–83. doi:10.1109/TITS.2020.3023957.
- Tuncali, C.E., Fainekos, G., Prokhorov, D., Ito, H., and Kapinski, J. (2020). Requirements-Driven Test Generation for Autonomous Vehicles With Machine Learning Components. *IEEE Transactions on Intelligent Vehicles*, 5(2), 265–280.

-
- UL (2022). UL4600: Standard for Safety for the Evaluation of Autonomous Vehicles and Other Products.
- Vagale, A., Bye, R.T., Oucheikh, R., Osen, O.L., and Fossen, T.I. (2021). Path planning and collision avoidance for autonomous surface vehicles II: a comparative study of algorithms. *Journal of Marine Science and Technology (Japan)*, 26(4), 1307–1323. doi:10.1007/s00773-020-00790-x.
- Vasstein, K., Brekke, E.F., Mester, R., and Eide, E. (2020). Autoferry Gemini: A real-time simulation platform for electromagnetic radiation sensors on autonomous ships. *IOP Conference Series: Materials Science and Engineering*, 929(1). doi:10.1088/1757-899X/929/1/012032.
- Veitch, E. and Alsos, O. (2022). A systematic review of human-AI interaction in autonomous ship systems. *Safety Science*, 152(May 2021). doi:10.1016/j.ssci.2022.105778.
- Wilthil, E.F., Flåten, A.L., and Brekke, E.F. (2017). A target tracking system for ASV collision avoidance based on the PDAF. *Lecture Notes in Control and Information Sciences*, 474(November), 269–288. doi:10.1007/978-3-319-55372-6_{_}13.
- Witsenhausen, H.S. (1966). A Class of Hybrid-State Continuous-Time Dynamic Systems. *IEEE Transactions on Automatic Control*, 11(2). doi:10.1109/TAC.1966.1098336.
- Woodcock, J., Larsen, P.G., Bicarregui, J., and Fitzgerald, J. (2009). Formal methods: Practice and experience. *ACM Computing Surveys*, 41(4). doi:10.1145/1592434.1592436.
- Yan, R., Yao, X., Yang, J., and B, K.H. (2018). Formal Collision Avoidance Analysis for Rigorous Building of Autonomous Marine Vehicles. In *Embedded Systems Technology*, 118–127. Singapore.

Part II

Selected Publications

Paper A:


Automatic simulation-based testing of autonomous ships using Gaussian processes and temporal logic

Tobias Rye Torben, Jon Arne Glomsrud, Tom Arne Pedersen, Ingrid B. Utne and Asgeir J. Sørensen

Journal of Risk and Reliability
Special Issue: Autonomous Systems Safety, Reliability, and Security
doi: [10.1177/1748006X211069277](https://doi.org/10.1177/1748006X211069277)

Automatic simulation-based testing of autonomous ships using Gaussian processes and temporal logic

Proc IMechE Part O:
J Risk and Reliability
1–21
© IMechE 2022
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/1748006X211069277
journals.sagepub.com/home/pio



Tobias Rye Torben¹ , Jon Arne Glomsrud², Tom Arne Pedersen²,
Ingrid B Utne¹  and Asgeir J Sørensen¹

Abstract

A methodology for automatic simulation-based testing of control systems for autonomous vessels is proposed. The work is motivated by the need for increased test coverage and formalism in the verification efforts. It aims to achieve this by formulating requirements in the formal logic Signal Temporal Logic (STL). This enables automatic evaluation of simulations against requirements using the STL robustness metric, resulting in a robustness score for requirements satisfaction. Furthermore, the proposed method uses a Gaussian Process (GP) model for estimating robustness scores including levels of uncertainty for untested cases. The GP model is updated by running simulations and observing the resulting robustness, and its estimates are used to automatically guide the test case selection toward cases with low robustness or high uncertainty. The main scientific contribution is the development of an automatic testing method which incrementally runs new simulations until the entire parameter space of the case is covered to the desired confidence level, or until a case which falsifies the requirement is identified. The methodology is demonstrated through a case study, where the test object is a Collision Avoidance (CA) system for a small high-speed vessel. STL requirements for safety distance, mission compliance, and COLREG compliance are developed. The proposed method shows promise, by both achieving verification in feasible time and identifying falsifying behaviors which would be difficult to detect manually or using brute-force methods. An additional contribution of this work is a formalization of COLREG using temporal logic, which appears to be an interesting direction for future work.

Keywords

Verification, autonomous vessels, Gaussian processes, temporal logic, COLREG

Date received: 6 April 2021; accepted: 8 December 2021

Introduction

With the rapid advances in the field of information and communication technology (ICT) in the recent years, the tasks that are automated gradually become more complex. Space and underwater operations with limited means of communication, as well as autonomous transportation solutions have been driving forces for this development. The autonomy trend has also reached the maritime sector, where several commercial and academic projects aiming toward autonomous maritime vessels have been announced recently.¹

Building a compelling argument for the safety of autonomous systems has proven to be a challenge.² In the maritime domain, extensive use of *simulation-based testing* has been proposed as a possible approach.³ Simulation-based testing refers to creating a simulation model, often termed a *digital twin*, of a system together

with its operative environment and performing testing on the digital twin instead of the actual system in the real world. Simulation-based testing is attractive due to its scalability, that is, it is possible to assess system level behaviors for highly complex systems. Autonomous vessel concepts are characterized by high levels of complexity in their hardware and software systems, as well as in their interaction with the operative environment.

¹Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology (NTNU), Trondheim, Norway
²Group Research and Development, Det Norske Veritas (DNV), Høvik, Norway

Corresponding author:

Tobias Rye Torben, Centre for Autonomous Marine Operations and Systems, Norwegian University of Science and Technology (NTNU), Otto Nielsens vei 10, Trondheim 7052, Norway.
Email: tobias.torben@ntnu.no

Combined with the intrinsic challenges related to verification of autonomous functions, such as the use of machine learning components and hard-to-predict emergent behaviors, simulation-based testing stands out as a promising and key way forward. Simulation-based testing, in particular Hardware-in-the-Loop (HiL) testing, has strong traditions in the maritime industry already.⁴⁻⁷

While simulation-based testing offers a great platform for verification, it is paramount that it is used in combination with valid processes for test case selection, evaluation of results and test coverage assessment. Traditionally, the selection of test cases has been a manual process based on risk analyses and experience with incidents and typical pitfalls in development and implementation. For emerging technologies, such as autonomous vessels, this approach is challenging, as the necessary experience, regulations, class notations, and best-practices are not yet available. Moreover, the operative environment for autonomous systems is characterized as highly dynamic, unstructured, and uncertain which gives a wide span of possible situations and failure combinations.⁸ This necessitates a large number of simulations to obtain sufficient test coverage, which calls for automation of the test case selection and corresponding evaluation of the results. Also, since simulation-based testing, in contrast to more formal methods, is almost never able to test all possible scenarios due to practical computation time constraints, the notion of *confidence level* in the verification becomes important. By confidence level, we refer to the probability that a verified system actually contains no requirement violations. Since autonomous ships are safety-critical, it is crucial to have methods to assess the confidence level in the verification efforts to build sufficient trust.

The main scientific contribution of our work is the development of a methodology for simulation-based testing which attempts to address the needs specified above. We propose to formulate the requirements to test against in the formal logic STL. This enables automatic quantitative evaluation of simulations against the given requirements. Furthermore, we define parametric test cases, where a more general parametric case is defined manually, and we verify that all concrete sub-cases meet the requirements. We use a Gaussian Process (GP) model to predict the performance and uncertainty level over the entire parameter space. The GP model is updated by running simulations and observing the resulting performance, and its estimates used to adaptively guide the test case selection toward cases with low performance or high uncertainty. The proposed testing method incrementally runs new simulations until the entire parameter space of the test case is covered to the desired confidence level, or until a case which falsifies the requirement is identified.

The proposed method has several advantages. We get an assessment of the coverage and quantification of the confidence in the verification effort. Due to the

adaptive test case selection, it is expected to reduce the number of simulations required to obtain a sufficient test coverage compared to a regular grid search. Efficient falsification is also achieved if the system does not meet the requirements. After an initial setup, the testing is completely automatic which enables the execution of a large number of simulations. This also integrates well with agile development and continuous deployment, where nightly builds can run automated simulation-based testing along with the standard unit and integration software tests. We also highlight that the proposed approach is not limited to using STL requirements and STL robustness evaluations. Any quantitative evaluation of simulations may be used instead. The core methodology is also by no means restricted to the testing of autonomous vessels, although that is the focus of this paper.

There exist several previous works on the topic of using STL and STL robustness in simulation-based testing. Prominent frameworks are Breach,⁹ Taliro,¹⁰ and RRT-REX.¹¹ The Taliro framework is used in Tuncali et al.¹² to do simulation-based falsification for autonomous driving. To our knowledge, no previous work exists which uses STL in combination with a GP model for verification. However, the methods have previously been combined to achieve data-driven synthesis of requirements.¹³

There also exist previous works in automatic evaluation of maritime Collision Avoidance (CA) systems for autonomous ships, including Woerner,¹⁴ which uses tailored penalty functions. This approach is further developed by Pedersen et al.³ Stankiewicz and Mullins¹⁵ propose a different approach by running a large number of simulations and mapping decision boundaries using clustering methods. Lee et al.¹⁶ present a falsification method called Adaptive Stress Testing for aircraft collision avoidance, which uses an adversary reinforcement learning based agent for the environment to identify falsifying interactions. The use of formal methods for specification and verification of autonomous systems has seen some adoption in the sectors of aerospace, automotive, and mobile robotics.¹⁷ During the last year, the maritime sector has also seen some use of formal methods for autonomous vessels. Shokri-Manninen et al.¹⁸ have created a formal automata-based model of single-vessel encounters and synthesized a correct-by-construction navigation strategy in the tool UPPAAL STRATEGO, which uses a combination of model checking and machine learning. Park and Kim¹⁹ has synthesized a correct-by-construction controller for automatic docking of marine vessels based on reachability analysis. Finally, Foster et al.²⁰ present a controller for autonomous marine vessels in the form of a hybrid dynamical system and use the Isabelle theorem prover to verify some invariant properties.

This paper is organized as follows. First, the background and mathematical preliminaries are presented. This includes an overview of state-of-the-art verification of cyber-physical systems (CPSs) and an

Table I. Explanation of symbols that are used extensively throughout the paper.

\mathbf{p}	List of parameter values which defines a simulation
\mathbf{w}	Output signal from a simulation
φ	STL formula
$\varphi(\mathbf{w}, t)$	STL robustness of signal \mathbf{w} at time t with respect to formula φ
f_p	The function which maps a list of parameters to an STL robustness score
\mathbf{P}	List of test points for the GP
ρ	List of random variables for the STL robustness at the test points
\mathbf{P}_{obs}	List of observed points for the GP
ρ_{obs}	List of random variables for the STL robustness at the observed points
$\bar{\rho}(\mathbf{p})$	GP mean estimate of the STL robustness score at point \mathbf{p}
$var(\rho(\mathbf{p}))$	GP variance estimate of the STL robustness score at point \mathbf{p}

introduction to temporal logic and GPs. Next, the main contribution, a methodology for automatic testing, is developed, and an implementation is presented in algorithmic form in Algorithm 1. A case study which demonstrates the use of the proposed methodology for a CA system is conducted thereafter. A discussion on methodical issues with the proposed approach follows next before concluding remarks are given.

Preliminaries and background

Terminology and definitions

In this section we define the main terminology and definitions used in the paper. A list of symbols that are used extensively throughout the paper is given in Table I. Bold symbols are used for vectors or matrices.

Suppose that we have a simulator of the system under test together with its operational environment. The result of a particular simulation depends on a number of parameters describing for instance the initial conditions, input signals, and configurations. For a simulator with n parameters, each parameter p_i , where $i \in [1, 2, \dots, n]$ is an index, has an associated *parameter set* \mathcal{P}_i which specifies the values that the parameter can take. The set \mathcal{P} , defined by the Cartesian product $\mathcal{P} = \mathcal{P}_1 \times \mathcal{P}_2 \times \dots \times \mathcal{P}_n$ contains all possible combinations for the simulator. It is usually not feasible nor useful to test all simulations in \mathcal{P} . Instead, we select some subsets where most of the parameters are fixed but some are allowed to vary within a set. We formalize this idea with the notion of a *case*. A case Σ is defined by a collection of k parameters p_1, p_2, \dots, p_k with corresponding parameter sets $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_k$. The parameter set of a case Σ is defined as the Cartesian product of the parameter sets of all of the parameters in the case, $\mathcal{P}_\Sigma = \mathcal{P}_1 \times \mathcal{P}_2 \times \dots \times \mathcal{P}_k$. Cases are arranged hierarchically according to their parameter sets. We say that case Σ_1 is a *sub-case* of Σ_2 if $\mathcal{P}_{\Sigma_1} \subset \mathcal{P}_{\Sigma_2}$. Similarly, Σ_2 is a *super-case* of Σ_1 if $\mathcal{P}_{\Sigma_1} \subset \mathcal{P}_{\Sigma_2}$.

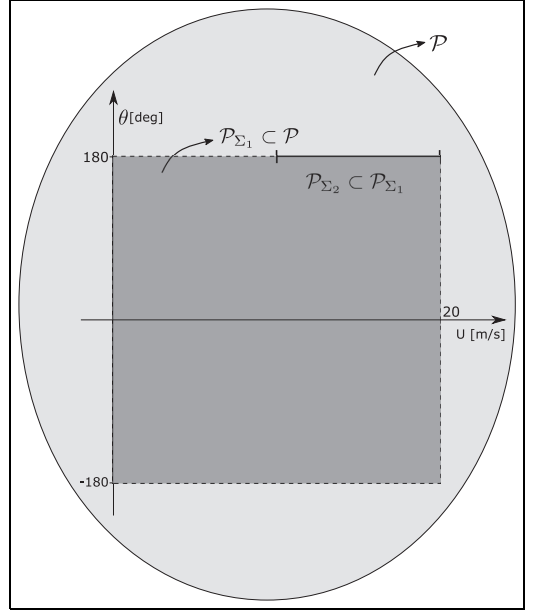


Figure 1. Illustration of the parameters sets in Example 1. \mathcal{P} is the full parameter set for the simulator; here projected in 2D for visualization. \mathcal{P}_{Σ_1} is the 2D subset of \mathcal{P} corresponding to the case Σ_1 , and \mathcal{P}_{Σ_2} is the 1D subset of \mathcal{P}_{Σ_1} corresponding to the case Σ_2 .

Each point $\mathbf{p} \in \mathcal{P}$ represents a list of parameter values. This defines the input to a simulation. For a particular simulator, we define a simulation as a function $sim : \mathcal{P} \mapsto \mathbf{W}$, which maps a vector of parameter values, $\mathbf{p} \in \mathcal{P}$ to a time stamped output signal $\mathbf{w} = (y_0, t_0), (y_1, t_1), \dots, (y_N, t_N)$. The output signal vector \mathbf{y} has m components and its domain is the set $\mathcal{Y} = \mathcal{Y}_1 \times \mathcal{Y}_2 \times \dots \times \mathcal{Y}_m$.

We demonstrate these definitions in the following example.

Example 1. Consider a simulator describing an autonomous marine vessel at open sea. Such a simulator will likely have a large number of configurable parameters. Suppose that we want to verify that the autonomous vessel can handle all situations where another is on direct collision course at different speeds. This can be described by a case Σ_1 with two parameters, the course of the other vessel, $\theta \in [-180, 180]^\circ$ and the speed of the other vessel, $U \in [0, 20]m/s$, and all other parameters remain fixed. The parameter set of Σ_1 , $\mathcal{P}_{\Sigma_1} = [-180, 180] \times [0, 20]$ is a two-dimensional section of \mathcal{P} . Next, suppose that we want to examine head-on situations with high speed more closely. This can be described by a sub-case $\Sigma_2 \subset \Sigma_1$, where the course is fixed at $\theta = 180^\circ$ and $U \in [10, 20]m/s$. This is illustrated in Figure 1, which shows \mathcal{P}_{Σ_1} as a two-dimensional subset of \mathcal{P} , and \mathcal{P}_{Σ_2} as a one-dimensional subset of \mathcal{P}_{Σ_1} .

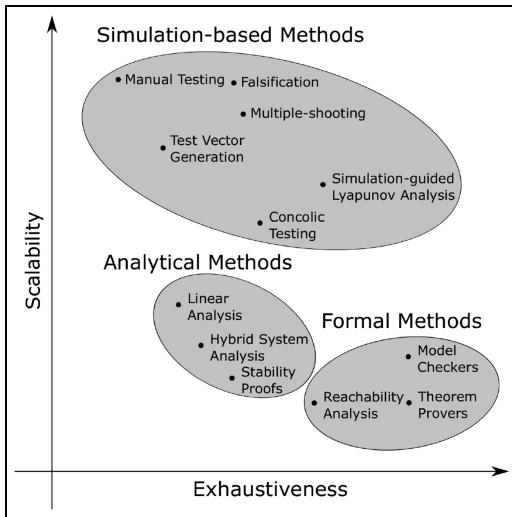


Figure 2. Spectrum of verification methods for CPSs. The horizontal axis indicates the level of exhaustiveness/formality. The vertical axis indicates the scalability of the method, that is, how large or complex systems it can verify. Inspired by Kapinski et al.²¹

Verification methods for Cyber-Physical Systems

Cyber-Physical Systems (CPSs) are comprised of physical, digital, and networking components. Typically a physical plant is controlled by digital computers. Autonomous vessels are clearly an example of this. Since the invention of the micro processor, CPSs have become ubiquitous. The dual nature of these systems, including both hardware and software, introduces challenges in the verification of them, and this has been an active area of research and development. Here, we give a brief overview of existing methods and their merits and shortcomings to build a context around the proposed method of this paper.

In Figure 2, a classification of the different verification methods is shown. The methods are ranked informally according to their scalability, that is, how large or complex systems they can verify, and the exhaustiveness, which is an indication of how thoroughly the possible behaviors of a system are assessed. Figure 3 shows the corresponding system models used in the verification. The models are ranked according to how closely they can resemble the real system that is verified. The methods can be classified into three distinct classes; formal, analytical, and simulation-based.

Formal methods are characterized by a high level of formality and exhaustiveness. They are usually based on a finite transition system model, which is a discrete model containing a finite number of states and a finite number of transitions between states. Formal methods are usually used in combination with a formal

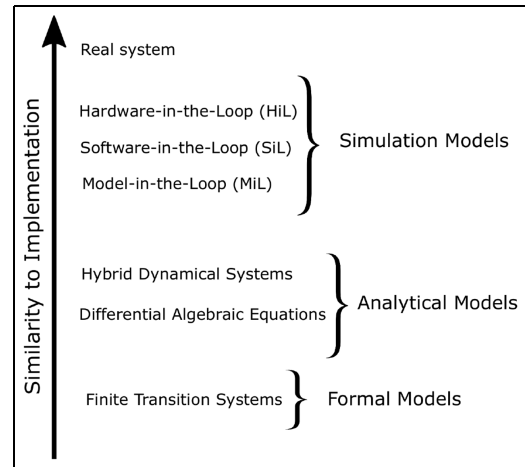


Figure 3. Spectrum of models used in the verification process, ranked according to how close the models can come to the actual implementation of the CPS.

specification language or logic, such as temporal logic, to specify the desired behavior. Model checking is the most common formal method, which does an exhaustive search of all of the possible executions of a finite transition system to verify that it satisfies a temporal property.²² Reachability analysis approximates the set of behaviors that a system can exhibit, and can for instance be used to verify that a set of unsafe behaviors is never visited.²³ Theorem provers are computer tools which aid the user in building mathematical proofs that a model meets a property.²⁴ There exist both interactive theorem provers, which require input from the user for each step of the proof, and fully automatic theorem provers.

Model checkers and reachability tools have seen limited scalability as they have been limited by the so-called *state-space explosion* problem. This refers to the fact that the set of states grows exponentially as the number of variables increases. However, with the huge increase in computational power and advances in the model checking algorithms, modern model checkers can solve highly complex verification problems.

Most formal methods are based on some form of finite-transition system, which also needs to be limited in size due to the state-space explosion problem.¹⁷ Therefore it is sometimes required to create a separate verification model suitable for formal verification, which is an abstraction of the implementation. Since the model which is verified is an abstraction of the implementation, this creates a *reality gap*,¹⁷ which may miss some implementation-level errors. Building an implementation based on a formally verified design has nevertheless proven to be a successful strategy for uncovering fundamental errors in the system. The reality gap is particularly evident for systems with continuous dynamics, which need to be discretized into a

finite-transition model to facilitate model checking. Theorem provers are often better able to verify properties of systems with continuous dynamics. For software systems, there exist formal verification tools which can work directly on implementation-level code. There is clearly a wide spectrum of modeling power at play here. However, compared to for example, simulation-based methods, the models suitable for formal verification generally have stronger limitations with respect to the type and complexity of systems that they can model, and therefore score lower on similarity to implementation.²¹

Formal methods have also proven to offer great advantages in the development process, by enabling the development of correct-by-construction designs built on formal specifications. Taking on a formal approach early in the development process can also improve the verification process later, by eliminating many classes of errors and by having a formal specification to verify against.²⁵ For an in-depth survey of this topic the reader is referred to Luckuck et al.¹⁷

Analytical methods perform manual mathematical analysis on a set of symbolic equations. For continuous systems, the models are Differential Algebraic Equations (DAEs), whereas systems which exhibit both continuous and discrete behaviors are modeled as hybrid dynamical systems. These models rank quite low on similarity to implementation due to the sheer difficulty in creating analytical models of complex CPSs. Stability proofs refer to using mathematical analysis, such as Lyapunov methods to prove for example, stability or invariance of a set for continuous systems.²⁶ Linear Analysis is similar, but here the system is first linearized about an operational point. This is quite trivial to do even for complex models however, the results are only local and thus not very exhaustive.²⁷ Linear analysis is also possible to do numerically for some simulation models. Hybrid system analysis refers to a set of mathematics studying the properties of hybrid dynamical systems, and is able to handle more complex systems than pure DAEs, since it supports discrete dynamics.²⁸ The mathematical theorems mostly resemble those of DAEs. However, the results which can be proved for hybrid dynamical systems are often not as strong as those of DAEs because they are more complex mathematically, and their theoretical foundation is still relatively immature.

Simulation-based methods use numerical simulation models which can be stepped forward in time using numerical solvers. We have divided simulation models for CPSs into three classes. In a Model-in-the-Loop (MiL) simulation, both the controller and plant are simulation models. In a Software-in-the-Loop (SiL) simulation, the plant is controlled by the real control software, and in HiL simulation the control system runs on the real hardware platform and communicates with the simulated plant through a HiL interface. It is possible to create detailed simulation models of large and complex CPSs, even entire ships or aircraft, which

together with the HiL and SiL opportunities make it both highly scalable and close to the real implementation.²¹ The reality gap is present also for simulation-based methods, as they operate on models of reality. In particular, complex environments can be hard to model accurately. Nevertheless, due to the strong modeling power and flexibility of simulation-models, this is less pronounced than for other analytical and formal methods.

Simulation is inherently a testing approach, as a single simulation only assesses behavior for a single test case. Manual simulation-based testing therefore scores low on exhaustiveness. The other simulation-based methods in Figure 2 represent different approaches to improve on this by systematically managing the test case selection and evaluation of the results in various ways. Test vector generation is an automated process for selecting the test cases such that certain coverage criteria are met. Concolic testing combines simulations of the plant with a formal analysis of the decision branching of the controller software. Falsification methods use a parameterization of test cases and take an optimization approach to search for cases with low performance. Multiple-shooting approaches also attempt to achieve falsification by running many partial simulations and splicing together the results to identify a falsifying case. Finally, simulation-guided Lyapunov analysis refers to a method for searching for a Lyapunov function using the results of simulations. From a Lyapunov function, stability, invariant sets, and performance bounds can be derived. For a more detailed description of these methods, the reader is referred to Kapinski et al.²¹

The method proposed in this paper falls into the class of simulation-based methods which aim to increase the exhaustiveness and formality. It may be used with any of the simulation models in Figure 3.

Temporal logic

Temporal logics are extensions of propositional logic which also capture temporal aspects. A temporal logic formula specifies a behavior, and there exists effective algorithms to evaluate a signal against a temporal logic formula to see if it satisfies the specified behavior. Linear Temporal Logic (LTL), was introduced in Pnueli.²⁹ LTL operates on Boolean signals in discrete time. An extension of LTL is Metric Temporal Logic (MTL) which operates on Boolean signals in continuous time.³⁰ Signal Temporal Logic (STL) was proposed as a syntactic addition to MTL, where the formulas operate on real-valued signals.³¹

During the last decade, many have realized the power and possibilities when specifying behaviors in STL, see Kapinski et al.²¹ and the references therein. In addition to formal specification and verification, it can be used as a runtime monitor,³² where the online behavior is continuously evaluated against an STL requirement. Applications of this include encoding traffic rules

as STL and using this to monitor road safety online.³³ Moreover, it has also seen several applications as a design methodology in planning³⁴ and control synthesis.³⁵

The basic building block of STL formulas are *predicates*. A predicate π is a function which maps a signal y to a Boolean. In STL, the predicates take the form

$$\pi ::= f(y) \leq c, \quad (1)$$

where $f(y)$ is a scalar, real-valued function which maps the input y signal to a real-valued scalar, and c is a real-valued scalar. The signals which satisfy a predicate, π , represent a subset in the space \mathcal{Y} . In the following, we represent the set that corresponds to the predicate π using the notation $\mathcal{O}(\pi)$. We note that STL predicates are only defined for non-strict inequalities. However, since we will evaluate the formulas using the quantitative STL robustness semantics, there is no difference between strict and non-strict inequalities.

STL formulas are built by combining predicates with the operators of propositional logic and temporal operators. An informal description of the various operators is given before the formal syntax and semantics of STL are presented.

- *Conjunction*: $\varphi_1 \wedge \varphi_2$ is true if both φ_1 and φ_2 are true.
- *Disjunction*: $\varphi_1 \vee \varphi_2$ is true if either φ_1 or φ_2 are true.
- *Negation*: $\neg\varphi$ is true if φ is false.
- *Implication*: $\varphi_1 \rightarrow \varphi_2$ is true if φ_2 follows from φ_1 , that is, $\varphi_1 \rightarrow \varphi_2$ is false if and only if φ_1 is true and φ_2 is false.
- *Eventually*: $\diamond\varphi$ is true if φ is true at some time.
- *Always*: $\square\varphi$ is true if φ is true at all times.
- *Next*: $\bigcirc\varphi$ is true if φ is true at the next discrete time step.
- *Until*: $\varphi_1 U \varphi_2$ is true if φ_1 is true until φ_2 first becomes true.
- *Release*: $\varphi_1 R \varphi_2$ is true if φ_2 is true until φ_1 first becomes true.

Definition 1. *STL Syntax*³¹:

Let Π be the set of predicates and \mathcal{I} be any non-empty connected interval of $\mathbb{R}_{\geq 0}$. The set of well-formed STL formulas is defined by the grammar

$$\varphi ::= \text{T} \mid \pi \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \bigcirc\varphi \mid \varphi_1 U_{\mathcal{I}}\varphi_2, \quad (2)$$

where φ , φ_1 , and φ_2 are STL formulas, T is the True constant, and $\pi \in \Pi$ is an STL predicate.

Note that all the logical and temporal operators can be derived from these basic operators:

$$\varphi_1 \wedge \varphi_2 \equiv \neg(\neg\varphi_1 \vee \neg\varphi_2) \quad (3)$$

$$\varphi_1 \rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2 \quad (4)$$

$$\diamond_{\mathcal{I}}\varphi \equiv \text{T} U_{\mathcal{I}}\varphi \quad (5)$$

$$\square_{\mathcal{I}}\varphi \equiv \neg\diamond_{\mathcal{I}}\neg\varphi \quad (6)$$

$$\varphi_1 R_{\mathcal{I}}\varphi_2 \equiv \neg(\neg\varphi_1 U_{\mathcal{I}}\neg\varphi_2) \quad (7)$$

Example 2. An example of a simple STL formula, which will be used extensively in this paper, is the requirement for a vessel to always keep a safe distance from other vessels. In STL this can be expressed as

$$\varphi_{\text{safety}} = \square\neg(d(t) \leq d_{\min}) \quad (8)$$

where $d(t)$ is the distance to another vessel at time t and d_{\min} is a constant specifying the minimum required safety distance.

STL robustness metric

Much of the popularity of STL is due to the *STL robustness metric*, introduced by Fainekos and Pappas.³⁶ In contrast to the normal Boolean semantics, which only give a true/false evaluation of whether a signal satisfies a formula, the STL robustness metric defines the semantics for quantitative evaluation of how robustly a signal satisfies an STL formula. Hence, STL provides both a language to describe behaviors and a metric to measure conformance to these behaviors. This has proven to be a powerful combination.³⁷

Let $\llbracket\varphi\rrbracket(\mathbf{w}, t)$ denote the STL robustness of a signal \mathbf{w} against the formula φ at a discrete time t . The STL robustness metric has the soundness property that $\llbracket\varphi\rrbracket(\mathbf{w}, t) \geq 0$ if \mathbf{w} satisfies φ at time t and $\llbracket\varphi\rrbracket(\mathbf{w}, t) < 0$ otherwise. Informally, the magnitude of the robustness metric indicates how much the signal can change without violating the requirement.

Before we formally define the STL robustness metric we must first define *metrics* and the *signed distance*.

Definition 2. *Metric*³⁶:

A metric on a set S is a positive function $d : S \times S \rightarrow \mathbb{R}_{\geq 0}$ such that

$$(1) \quad \forall s, s' \in S, \quad d(s, s') = 0 \Leftrightarrow s = s'$$

$$(2) \quad \forall s, s' \in S, \quad d(s, s') = d(s', s)$$

$$(3) \quad \forall s, s', s'' \in S, \quad d(s, s'') \leq d(s, s') + d(s', s'')$$

In this paper the *Euclidean metric* $d(s, s') = \|s - s'\|$; has been used.

Next, we define the *signed distance* to a set. Intuitively, this captures how robustly a point belongs to a set. If the point is in the set, the signed distance is

positive. If the point is on the boundary then it is zero and if the point is outside the set it is negative. The magnitude represents how far away the point is from the boundary. A formal definition follows.

Definition 3. *Signed distance*³⁶:

Consider a point $s \in S$ a set $A \subseteq S$ and a metric d on S . The signed distance from s to A is defined as

$$Dist_d(s, A) := \begin{cases} -\inf\{d(s, s') | s' \in A\} & \text{if } s \notin A \\ \inf\{d(s, s') | s' \notin A\} & \text{if } s \in A \end{cases} \quad (9)$$

With the definitions of metrics and the signed distance, a formal definition of the STL robustness semantics is stated next.

Definition 4. *STL robustness semantics*³⁶:

For a metric d and a signal $\mathbf{w} = (y_0, t_0), (y_1, t_1), \dots, (y_N, t_N)$, the STL robustness $\llbracket \varphi \rrbracket_d(\mathbf{w}, t)$ of \mathbf{w} w.r.t φ at time instance $t \in [0, 1, \dots, N]$ is defined as:

$$\llbracket \mathbf{T} \rrbracket_d(\mathbf{w}, t) := +\infty \quad (10a)$$

$$\llbracket \pi \rrbracket_d(\mathbf{w}, t) := Dist_d(y_t, \mathcal{O}(\pi)) \quad (10b)$$

$$\llbracket \neg \varphi \rrbracket_d(\mathbf{w}, t) := -\llbracket \varphi \rrbracket_d(\mathbf{w}, t) \quad (10c)$$

$$\llbracket \varphi_1 \vee \varphi_2 \rrbracket_d(\mathbf{w}, t) := \max(\llbracket [0, 1] \rrbracket_d(\mathbf{w}, t), \llbracket \varphi_2 \rrbracket_d(\mathbf{w}, t)) \quad (10d)$$

$$\llbracket \bigcirc \varphi \rrbracket_d(\mathbf{w}, t) := \begin{cases} \llbracket \varphi \rrbracket_d(\mathbf{w}, t+1) & \text{if } t+1 \in N \\ -\infty & \text{otherwise} \end{cases} \quad (10e)$$

$$\begin{aligned} \llbracket \varphi_1 U_{\mathcal{I}} \varphi_2 \rrbracket_d(\mathbf{w}, t) &:= \max_{j.s.t. (t_j - t_i) \in \mathcal{I}} \\ &\left(\min \left(\llbracket \varphi_2 \rrbracket_d(\mathbf{w}, j), \min_{t \leq k < j} \llbracket \varphi_1 \rrbracket_d(\mathbf{w}, k) \right) \right), \end{aligned} \quad (10f)$$

where $\pi \in \Pi$ is a predicate, and $\mathcal{O}(\pi)$ is the corresponding set in \mathcal{Y} . For short, $\llbracket \varphi \rrbracket(\mathbf{w})$ refers to the robustness at time $t = 0$ using the Euclidean norm as the metric.

Example 3. As an example, we illustrate the use of the STL robustness metric for the safety distance requirement of (8), as shown in Figure 4. A sample signal for the distance between two vessels $d(t)$ has been generated by a random Wiener process. In the top plot, $d(t)$ is plotted together with the minimum safety distance limit $d_{min} = 50m$. This is the input signal to an STL monitor, whose output is shown in the lower plot. To build an intuitive understanding of the resulting STL robustness score, we split the evaluation of the signal into two steps. First, we look at the inner subformula, $\varphi = \neg(d(t) \leq d_{min})$. This STL formula is time-independent, that is, the value of $\llbracket \varphi \rrbracket(d(t), t)$ only depends on the value of $d(t)$ at time t . This robustness value, shown in yellow, intuitively represents how far the distance to the other vessel is from violating the requirement. This

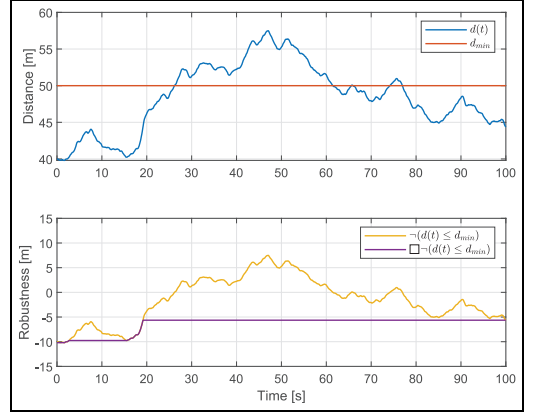


Figure 4. STL robustness evaluation against safety distance requirement. The upper plot shows a sample distance signal, $d(t)$ together with the required minimum distance d_{min} , plotted against time. The lower plot shows the corresponding STL robustness score for the subformula $\varphi = \neg(d(t) \leq d_{min})$ and the full formula $\varphi_{safety} = \square \neg(d(t) \leq d_{min})$.

means that when $d(t) = 50m$, $\llbracket \varphi \rrbracket(d(t), t) = 0m$. When $d(t) > 50m$, $\llbracket \varphi \rrbracket(d(t), t) > 0$ and when $d(t) < 50m$, $\llbracket \varphi \rrbracket(d(t), t) < 0$.

When adding an operator on this subformula, the robustness score of the subformula becomes the input signal for this operator. Since \square is a temporal operator, the robustness score at any time depends on the entire time series from that time to the end. In the case of the always operator, it can be shown that the output robustness score is the minimum of the input signal over time. The robustness score for the full formula is shown in violet in the lower plot of Figure 4. This demonstrates that the value of the violet curve at time t corresponds to the minimum of the yellow curve from t to the end.

Gaussian processes

Consider an unknown function, $y = f(\mathbf{x})$, which we want to estimate by making observations (\mathbf{x}_i, y_i) . The standard regression approach is to assume a model for $f(\mathbf{x})$, and try to find the parameters of this model such that it fits the observations well. GPs take a different approach to this. A GP models the function values $y_i = f(\mathbf{x}_i)$ at points \mathbf{x}_i as random variables which are jointly Gaussian distributed.³⁸ The covariance between function values at points \mathbf{x}_p and \mathbf{x}_q is denoted

$$\text{cov}(f(\mathbf{x}_p), f(\mathbf{x}_q)) = k(\mathbf{x}_p, \mathbf{x}_q), \quad (11)$$

where $k(\mathbf{x}_p, \mathbf{x}_q)$ is a *covariance function* of choice, also known as a *kernel function*. A common choice of covariance function is the squared exponential

$$k(\mathbf{x}_p, \mathbf{x}_q) = \sigma^2 \exp\left(-\frac{1}{2l^2} \|\mathbf{x}_p - \mathbf{x}_q\|^2\right), \quad (12)$$

which is built on the assumption that points that are close to each other are more strongly correlated than points which are far apart. This function has two parameters: the variance σ^2 , and the length scale l . Note that for $\mathbf{x}_p = \mathbf{x}_q$ the covariance reduces to σ^2 which is the variance of the function value at this point. Informally, l describes how much $\|\mathbf{x}\|$ needs to change in order to significantly change the value of $f(\mathbf{x})$.

Further, assume that we can make uncertain observations of $f(\mathbf{x})$ by the measurement equation

$$y_{obs} = f(\mathbf{x}) + \epsilon, \quad (13)$$

where the measurement noise ϵ is normally distributed with zero mean and variance σ_ϵ^2 , and is independent from the noise of other observations.

Suppose that we have made observations of f at n_{obs} points, given by the random variables $\mathbf{y}_{obs} \in \mathbb{R}^{n_{obs}}$ and wish to predict the value of f at n test points, given by the random variables $\mathbf{y} \in \mathbb{R}^n$. It can be shown that the random variables $[\mathbf{y}_{obs}^T, \mathbf{y}^T]^T$ are jointly Gaussian distributed.³⁸ Their probability distribution is

$$\mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}(\mathbf{X}_{obs}, \mathbf{X}_{obs}) + \sigma_\epsilon^2 \mathbf{I} & \mathbf{K}(\mathbf{X}_{obs}, \mathbf{X}) \\ \mathbf{K}(\mathbf{X}, \mathbf{X}_{obs}) & \mathbf{K}(\mathbf{X}, \mathbf{X}) \end{bmatrix}\right), \quad (14)$$

where \mathbf{I} is the identity matrix. $\mathbf{K}(\mathbf{X}_{obs}, \mathbf{X})$ denotes the $n_{obs} \times n$ matrix of the covariances evaluated at all pairs of observations and test points, and similarly for the other entries $\mathbf{K}(\mathbf{X}_{obs}, \mathbf{X}_{obs})$, $\mathbf{K}(\mathbf{X}, \mathbf{X})$ and $\mathbf{K}(\mathbf{X}, \mathbf{X}_{obs})$.

To make predictions at the test points we take a Bayesian inference approach, and calculate the conditional probability distribution of $\mathbf{y}|\mathbf{y}_{obs}$. It can be shown³⁸ that this results in another Gaussian distribution with mean

$$\bar{\mathbf{y}} = \mathbf{K}(\mathbf{X}, \mathbf{X}_{obs}) [\mathbf{K}(\mathbf{X}_{obs}, \mathbf{X}_{obs}) + \sigma_\epsilon^2 \mathbf{I}]^{-1} \mathbf{y}_{obs}, \quad (15)$$

and covariance matrix

$$\begin{aligned} \text{COV}(\mathbf{y}) &= \mathbf{K}(\mathbf{X}, \mathbf{X}) - \\ &\mathbf{K}(\mathbf{X}, \mathbf{X}_{obs}) [\mathbf{K}(\mathbf{X}_{obs}, \mathbf{X}_{obs}) + \sigma_\epsilon^2 \mathbf{I}]^{-1} \mathbf{K}(\mathbf{X}_{obs}, \mathbf{X}). \end{aligned} \quad (16)$$

Hence, for each test point we have now obtained a Gaussian distribution with an associated mean and covariance. The mean value is used as the predicted values of the unknown function f at the test points. The diagonal elements of the covariance matrix represent the variance of the estimates, and can be used to establish confidence intervals on the predictions. This is illustrated in Figure 5.

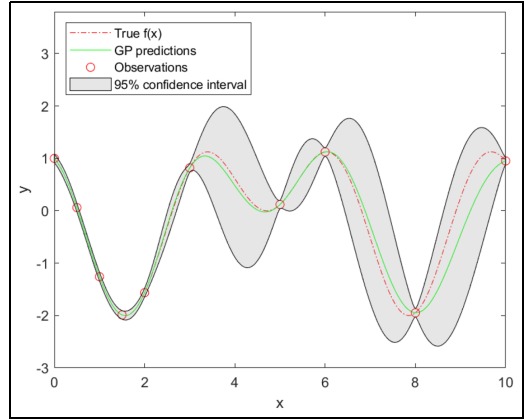


Figure 5. An example of a GP with 10 observations of $y = f(x)$.

The GP mean estimates at the test points are shown by the green curve together with their 95% confidence intervals. The figure shows how the GP fit is tighter for small values of x , where the observations are denser. For large values of x , observations are sparser. This gives a less accurate fit, which is reflected by the increased uncertainty bounds.

Automatic testing using Gaussian processes and temporal logic

In this section, we develop the main scientific result of the paper, a new method for automatic and adaptive simulation-based testing of autonomous vessels. An overview of how the simulator, STL monitor and GP model interact in our proposed method is given in Figure 6.

Problem statement and scope

Consider a case Σ with parameters in \mathcal{P}_Σ , and a requirement in the form of an STL formula. Recall that a simulation satisfies φ if and only if the STL robustness score is greater than 0. We define the confidence level of a verification as the probability that the STL robustness score is greater than zero for all points $\mathbf{p} \in \mathcal{P}_\Sigma$. The objective of the proposed method is to produce evidence that φ is satisfied with a probability greater than the desired confidence level p_{conf} for all points $\mathbf{p} \in \mathcal{P}_\Sigma$.

For simplicity, we restrict the parameter set \mathcal{P}_Σ to be a range set in \mathbb{R}^k . This can trivially be extended to cover simply connected sets, while the non-connected case can be handled by treating each connected subset separately.

Proposed approach

For a particular STL requirement, φ we propose to model the variations in the STL robustness for different choices of case parameters as a GP. Hence, we consider $f_\rho : \mathcal{P}_\Sigma \mapsto \mathbb{R}$ as an unknown function which maps a

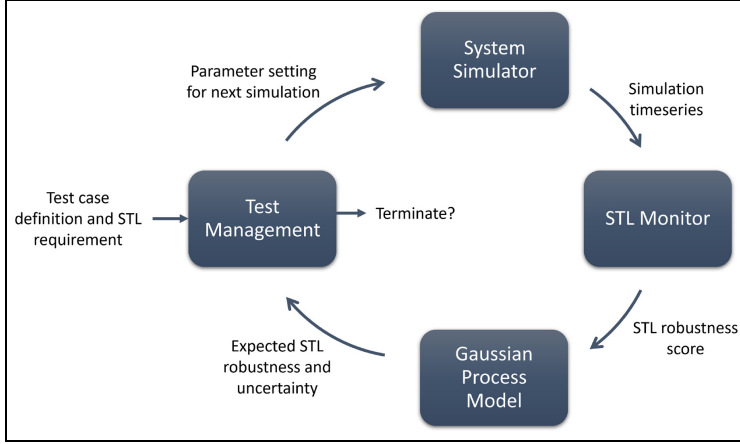


Figure 6. An overview of the main components of the proposed testing methodology and how they interact. The input to the method is a parametric test case definition and an STL requirement to test against. The test management selects a concrete test case from the test space and sends its parameter setting to the simulator. The resulting simulation time series is given as input to the STL monitor which calculates the STL robustness score. The GP model, which estimates the STL robustness score as a function of the test case parameters, is updated using the observed STL robustness score. The expected STL robustness and its uncertainty are used for smart selection of the next test case to simulate, and to determine if the test coverage is sufficient to terminate.

parameter vector $\mathbf{p} \in \mathcal{P}_\Sigma$ to a robustness value $\rho \in \mathbb{R}$. In reality, f_ρ is a known function defined by

$$f_\rho(\mathbf{p}) := \llbracket \varphi \rrbracket(\text{sim}(\mathbf{p})). \quad (17)$$

However, since evaluating this function requires running a simulation, it is often computationally intractable to evaluate it for all points $\mathbf{p} \in \mathcal{P}_\Sigma$, which makes GP estimation an attractive option. We estimate f_ρ at n test points in \mathcal{P}_Σ . These points are collected in the matrix $\mathbf{P} \in \mathbb{R}^{k \times n}$, such that each column in \mathbf{P} is a parameter vector \mathbf{p} , that is, a point in \mathcal{P}_Σ . The choice of points in \mathbf{P} is typically a uniform grid over \mathcal{P}_Σ with the desired resolution. To each test point we assign a random variable representing the unknown STL robustness value at this point. These are collected in the vector $\boldsymbol{\rho} \in \mathbb{R}^n$. We model our prior beliefs in the unknown function $f_\rho(\mathbf{p})$ by a joint Gaussian probability distribution with zero mean and covariance matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$. The (i, j) th entry in \mathbf{K} is given by some covariance (kernel) function $k(\mathbf{p}_i, \mathbf{p}_j)$.

Next, we wish to observe values of the STL robustness by running simulations and evaluating the resulting output signals against φ using the semantics of (10). We assume that a simulation gives us an uncertain observation of the robustness, given by (13). In reality, the outcome of a simulation is completely deterministic, however, adding a small uncertainty has some technical advantages. This is discussed more closely in the discussion section.

Taking a Bayesian inference approach, the posterior predictions of the robustness values at the points \mathbf{P} are

updated based on observed robustness values. Suppose that n_{obs} observations have been made at points $\mathbf{P}_{obs} \in \mathbb{R}^{k \times n_{obs}}$. The observed robustness values are collected in the vector $\boldsymbol{\rho}_{obs} \in \mathbb{R}^{n_{obs}}$. Given the assumptions made this far, the robustness values at the points \mathbf{P} and the observed points at \mathbf{P}_{obs} are jointly Gaussian with distribution

$$\begin{bmatrix} \boldsymbol{\rho}_{obs} \\ \boldsymbol{\rho} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I} & \mathbf{K}_{cross} \\ \mathbf{K}_{cross}^\top & \mathbf{K} \end{bmatrix}\right), \quad (18)$$

where $\mathbf{K}_{obs} \in \mathbb{R}^{n_{obs} \times n_{obs}}$ is the covariance matrix for the observation points and $\mathbf{K}_{cross} \in \mathbb{R}^{n \times n_{obs}}$ is the cross covariance between points in \mathbf{P} and \mathbf{P}_{obs} .

Using Bayesian inference, the conditional probability distribution of $\boldsymbol{\rho}$ given $\boldsymbol{\rho}_{obs}$ is given by

$$\begin{aligned} \boldsymbol{\rho} | \boldsymbol{\rho}_{obs} &\sim \mathcal{N}(\mathbf{K}_{cross}[\mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I}]^{-1} \boldsymbol{\rho}_{obs}, \\ &\mathbf{K} - \mathbf{K}_{cross}[\mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I}]^{-1} \mathbf{K}_{cross}^\top) \end{aligned} \quad (19)$$

Hence, at each point in \mathbf{P} we now have an associated expected value and uncertainty. In the following, let the operators $\bar{\rho}(\mathbf{p})$ and $\text{var}(\rho(\mathbf{p}))$ refer to the expected value and variance of the STL robustness at the point \mathbf{p} .

The main loop of our proposed method consists of iteratively running new simulations and updating the GP. This is repeated until a desired confidence level is achieved. The criterion for a sufficient confidence level is that the minimum of the p_{conf} probability confidence interval of f_ρ is greater than 0. That is, we terminate the search in a *Verified* state when

$$\min_{\mathbf{p} \in \mathbf{P}} \bar{\rho}(\mathbf{p}) - n_{conf} \sqrt{\text{var}(\rho(\mathbf{p}))} > 0. \quad (20)$$

Here, n_{conf} is the number of standard deviations associated with the confidence level p_{conf} .

Achieving this, we have shown that the probability, with which the system satisfies the requirement φ , is greater than p_{conf} , given the assumptions above. If the search process finds a case with robustness lower than 0, the search will terminate prematurely in a *Falsified* state, with an associated counterexample that can be used for debugging and improving the system.

Since observing the robustness at a point involves running a simulation, it is usually a time consuming operation. Thus, it is paramount that sample points are well chosen as to only explore interesting regions of \mathcal{P}_Σ and thereby minimizing the number of simulation runs. We propose to choose the next sampling point based on two criteria:

- Explore points with low expected robustness.
- Explore points with large uncertainty.

This can be combined in the following selection process

$$\mathbf{p}_{next} = \text{argmin}_{\mathbf{p} \in \mathbf{P}} \bar{\rho}(\mathbf{p}) - \kappa \sqrt{\text{var}(\rho(\mathbf{p}))}, \quad (21)$$

where $\kappa \in \mathbb{R}_{\geq 0}$ is a parameter deciding the trade-off between visiting areas with low predicted robustness and visiting areas where there is large uncertainty. This is commonly referred to as the *exploration versus exploitation trade-off*.

Before starting the search, we first draw n_{seed} *seed points* from \mathbf{P} , run simulations and build an initial GP from these observations. This serves to build an initial model of the robustness landscape before starting the adaptive search using (21). To obtain the maximum amount of information about f_ρ from n_{seed} samples, a design of experiments approach is taken, using *latin hypercube sampling* (LHS).³⁹ In essence, LHS divides each parameter into n_{seed} slots, and thereby creates a grid of hypercubes over \mathcal{P}_Σ . Along each dimension, LHS chooses samples such that no two samples occupy a hypercube in the same row, column, height and so forth. Within each hypercube, the sampling is random.

This concludes the development of the proposed testing approach. The method is stated more concisely in Algorithm 1. For a MATLAB/Simulink implementation, the reader is referred to the open-source online repository.⁴⁰

Validation of Gaussian process model

One of the most attractive properties of the proposed methodology is that it gives a quantification of the confidence level in the verification. This comes in the form of a probability of exceeding a determined robustness

limit. However, this probability is based on a statistical model with several assumptions and parameters. It is paramount that these are justified or validated in order to trust the resulting probabilities. The most notable assumption is that the robustness values are jointly Gaussian distributed with covariance given by the selected kernel function. This is hard to justify a priori, since little is known about the robustness function in advance. It is therefore necessary to validate this assumption after running the algorithm. For cases with only one parameter, this can to a large extent be done by visual inspection of the predicted robustness function. However, for higher-dimensional cases, such visual intuition is more difficult. We therefore need quantitative metrics to validate the GP model. A common choice for this is the *marginal likelihood* $p(\boldsymbol{\rho}_{obs} | \mathbf{P}_{obs})$. This represents the likelihood of observing the robustness values $\boldsymbol{\rho}_{obs}$ for the points \mathbf{P}_{obs} under the given GP model, giving an indication of the goodness of the model fit.

The marginal likelihood is calculated by marginalizing over all values for ρ

$$p(\boldsymbol{\rho}_{obs} | \mathbf{P}_{obs}) = \int p(\boldsymbol{\rho}_{obs} | \boldsymbol{\rho}, \mathbf{P}_{obs}) p(\boldsymbol{\rho} | \mathbf{P}_{obs}) d\boldsymbol{\rho} \quad (22)$$

As all probability distributions in (22) are Gaussian, this integral has an analytical solution given by the log marginal likelihood

$$\begin{aligned} \log p(\boldsymbol{\rho}_{obs} | \mathbf{P}_{obs}) &= -\frac{1}{2} \boldsymbol{\rho}_{obs}^T (\mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I})^{-1} \boldsymbol{\rho}_{obs} \\ &- \frac{1}{2} \log |\mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I}| - \frac{n_{obs}}{2} \log 2\pi \end{aligned} \quad (23)$$

Next, we propose a more complete validation method for the GP model. After achieving a verification, the normalized error e_n for each observation $(\mathbf{p}, f_\rho(\mathbf{p}))$ can be calculated as

$$e_n(\mathbf{p}) = \frac{f_\rho(\mathbf{p}) - \bar{\rho}(\mathbf{p})}{\sqrt{\text{var}(\rho(\mathbf{p}))}} \quad (24)$$

It can be shown that if the assumptions of the GP hold, then $e_n \sim \mathcal{N}(0, 1)$. We propose to validate the method by calculating a reference robustness function by simulating a large number of cases covering the parameter space. Then, the automatic testing algorithm is run, and we calculate the resulting mean and variance for each point on the reference function. This enables the calculation of a large number of normalized errors. The distribution of the normalized errors can then be compared to its theoretical distribution $\mathcal{N}(0, 1)$. The normality assumption can be assessed using a normal probability plot.

The method above is useful for validating the method for this paper. However, it is also important

Algorithm 1: Automatic simulation-based testing

```

input: Test case  $\Sigma$  Requirement  $\varphi$  Kernel function
 $k(\mathbf{p}_1, \mathbf{p}_2)$  and Hyperparameters  $n_{conf}$   $\kappa$   $\sigma_\epsilon$ 
Select  $n$  points uniformly from  $\mathcal{P}_\Sigma$  store in  $\mathbf{P}$ ;
Calculate covariance matrix  $\mathbf{K}$  for points in  $\mathbf{P}$ ;
// Find robustness for seed points
Sample  $n_{seed}$  seed points from  $\mathbf{P}$  using Latin
Hypercube Sampling
for  $i = 1$  to  $n_{seed}$  do
  Run simulation  $\mathbf{w}_i = sim(\mathbf{p}_i)$ 
  Calculate robustness  $[\varphi](\mathbf{w}_i)$ 
  Append  $(\mathbf{p}_i, [\varphi](\mathbf{w}_i))$  to list of observed points
   $(\mathbf{P}_{obs}, \rho_{obs})$ 
end
// Main loop
for  $i = (n_{seed} + 1)$  to  $maxNumberofSimulations$  do
  Calculate covariance matrices  $\mathbf{K}_{obs}$  and  $\mathbf{K}_{cross}$ 
  Calculate expected value for each point in  $\mathbf{P}$ :
   $\bar{\rho} = \mathbf{K}_{cross} [\mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I}]^{-1} \rho_{obs}$ 
  Calculate variance for each point in  $\mathbf{P}$ :
   $var(\rho) = \mathbf{K} - \mathbf{K}_{cross} [\mathbf{K}_{obs} + \sigma_\epsilon^2 \mathbf{I}]^{-1} \mathbf{K}_{cross}^T$ 
  if  $\min_{\mathbf{p} \in \mathbf{P}} \bar{\rho}(\mathbf{p}) - n_{conf} \sqrt{var(\rho(\mathbf{p}))} > 0$  then
    return Verified
  end
  Select next sample point:
   $\mathbf{p}_i = argmin_{\mathbf{p} \in \mathbf{P}} \bar{\rho}(\mathbf{p}) - \kappa \sqrt{var(\rho(\mathbf{p}))}$ 
  Run simulation  $\mathbf{w}_i = sim(\mathbf{p}_i)$ 
  Calculate robustness if  $\varphi(\mathbf{w}_i)$ 
  if  $\rho(\varphi, \mathbf{w}_i) < 0$  then
    return Falsified with counterexample  $\mathbf{p}_i$ 
  end
  Append  $(\mathbf{p}_i, \varphi(\mathbf{w}_i))$  to list of observed points
   $(\mathbf{P}_{obs}, \rho_{obs})$ 
end
// Max number of simulations reached
return Inconclusive

```

for users of the method to validate their verification results. In this case, the validation scheme defeats the whole purpose of the method, as it requires a large number of simulations when creating the reference surface. However, it may be used in a cross-validation setting. Observations can be split into n folds, and the GP can then be trained on $n - 1$ of the folds and the normalized error can be calculated for the remaining fold. This process can be repeated such that each fold is left out of training, and we therefore obtain a normalized error for each observation. For cases with two or more parameters, this should give enough simulations to enable a statistical study of the normalized errors. Two important indicators are the mean and the standard deviation of normalized errors. If the mean is significantly less than zero, it indicates that the GP is overestimating the robustness. If the standard deviation is greater than one, it indicates that the GP is overly confident in its predictions. If such problems appear, this can give input to adjust the hyper parameters and rerun the automatic testing algorithm. Since most of the observations have already been made, this adds little extra time as the time for the GP inference is usually negligible compared to running the simulations.

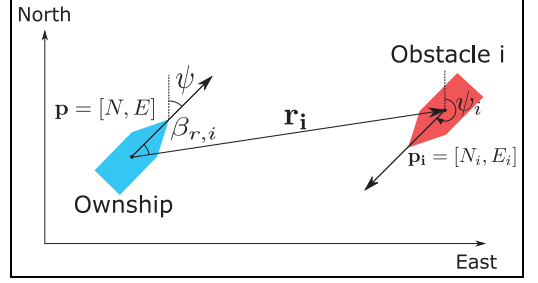


Figure 7. Definition of the symbols and notation used in the case study. The figure shows the position vector \mathbf{p} , heading ψ , relative position of obstacle i , \mathbf{r}_i , and relative bearing to obstacle i , $\beta_{r,i}$.

Case study: Open-sea collision avoidance

To demonstrate the use of the proposed method a case study has been conducted for CA in open sea. This section first presents the setup for the case study and the definition and evaluation of requirements. The results from applying the automatic testing method and STL evaluation to two test cases are presented and a statistical validation is performed.

Setup

The test object of the case study is the Branching Course Model Predictive Control (BC-MPC) algorithm. The system is implemented on a small high-speed vessel, details of the algorithm and the vessel are given in Eriksen et al.⁴¹ For the sake of simplicity, perfect situational awareness is assumed, that is, the CA system has perfect tracks on obstacles.

To evaluate simulations, certain signals are calculated based on position, heading, and speed of the vessels. The symbols and notation used in the case study are shown in Figure 7. It is assumed that the heading and course are identical, that is, no side-slipping or cross currents. The *distance at closest point of approach* (d_{CPA}) is defined as the smallest separation between two vessels if they continue with the current speed and course. The *time to closest point of approach* (t_{CPA}) is defined as the time until d_{CPA} occurs.

The STL robustness values for the requirements are saturated and normalized. This rationale for saturation is that very large robustness values are not interesting, but they make it harder to fit a GP to them. Normalization aims to ease the process of hyper parameter selection in the GP by having all requirements operate on the same scale, such that a set of hyper parameters are likely to work for a broad range of requirements. The normalization and saturation are defined by one extra parameter for each predicate, a normalization factor ν . The robustness of this predicate is calculated by first saturating the robustness to $[-\nu, \nu]$ and then dividing by ν such that all robustness values are mapped to the interval $[-1, 1]$.

Table 2. Parameters in the STL formulas and the normalization factors for the requirements. The subscript in the normalization factor corresponds to the signal that it normalizes.

Parameter	Value	Unit
$t_{CPA,turn}$	15	s
$d_{CPA,min}$	50	m
d_{min}	50	m
e_{max}	400	m
$v_{t_{CPA}}$	10	s
$v_{d_{CPA}}$	100	m
v_{β_r}	10	°
v_d	100	m
v_e	400	m

Requirements

Automatic quantitative evaluation of simulations are achieved by testing against three STL requirements: COLREG, safety distance and mission compliance. These are detailed in the following. All parameters used in the requirements are given in Table 2. We also emphasize that the automatic testing method scales well for testing against multiple requirements, as the GP for a particular requirement can be initialized with the observed simulations from previous runs against other requirements. It is also possible to test against all requirements in a single run, by testing against a conjunction of all requirements. The robustness for the individual requirements can be easily decomposed during post processing. For simplicity and clarity these optimizations are not utilized in this case study. The STL formulas for all requirements are summarized in Table 3.

COLREG requirement. COLREG are the international regulations for preventing collisions at sea.⁴² Automatic evaluation of COLREG compliance is a challenging topic due to the complexity and wide span of the possible scenarios combined with the inherent reliance on human judgment of the current COLREG. This section presents a new take on this problem, by using a formal language (STL) to express COLREG. Using a formal language enables systematic construction of modular requirements which can be subject to mathematical analysis such as consistency checking and formal verification. While this is by no means a complete system for COLREG evaluation, this aims to illustrate and motivate a new possible direction for automatic COLREG evaluation.

We propose to formulate the STL COLREG as a conjunction of reactive subformulas. In each subformula, the antecedent is a boolean variable expressing a COLREG situation, and the consequent is a set of required behaviors which must be satisfied in the corresponding COLREG situation. Five COLREG situations are defined, corresponding to COLREG Rules 13–17: *overtaking* (OT), *overtaken* (OV), *give way*

Table 3. Requirements and subformulas in the case study.

$\varphi_{safety} = \Box \neg (d_r \leq d_{min})$
$\varphi_{mission} = \Box (e \leq e_{max})$
$\varphi_{colreg} = \Box (HO \rightarrow \varphi_{HO} \wedge GW \rightarrow \varphi_{GW} \wedge OT \rightarrow \varphi_{OT})$
$\varphi_{HO} = t_{CPA} \leq t_{CPA,turn} \rightarrow \beta_r \in [-170^\circ, -10^\circ]$
$\varphi_{GW} = t_{CPA} \leq t_{CPA,turn} \rightarrow d_{CPA} \geq d_{CPA,min}$
$\varphi_{OT} = t_{CPA} \leq t_{CPA,turn} \rightarrow d_{CPA} \geq d_{CPA,min}$
$\varphi_{OT}^c = \varphi_{OV}^c = \varphi_{GW}^c = \varphi_{SO}^c = \varphi_{HO}^c = t_{CPA} \leq 0$
$\varphi_{NC}^c = d_{CPA} \leq d_{CPA,min} \wedge t_{CPA} \leq t_{CPA,min}$

(GW), *stand on* (SO), and *head-on* (HO). Here, only the situations which require explicit action by ownship are included in the requirement, given in (25).

$$\varphi_{colreg} = \Box (HO \rightarrow \varphi_{HO} \wedge GW \rightarrow \varphi_{GW} \wedge OT \rightarrow \varphi_{OT}) \quad (25)$$

In a head-on situation, COLREG require a port-to-port passing. This is enforced by requiring that the relative bearing β_r is between -170° and -10° when t_{CPA} is lower than the threshold value $t_{CPA,turn}$:

$$\varphi_{HO} = t_{CPA} \leq t_{CPA,turn} \rightarrow \beta_r \in [-170^\circ, -10^\circ] \quad (26)$$

For give way and overtaking situations, COLREG do not specify a required maneuver, but require that ownship makes early and substantial action to avoid a collision. We enforce this by requiring that d_{CPA} is larger than the threshold $d_{CPA,min}$ when the t_{CPA} is lower than the threshold $t_{CPA,turn}$:

$$\varphi_{GW} = \varphi_{OT} = t_{CPA} \leq t_{CPA,turn} \rightarrow d_{CPA} \geq d_{CPA,min} \quad (27)$$

The COLREG requirements of (25) use boolean signals which specify the type of COLREG situation. We adopt the COLREG situation selection of Tam and Bucknall,⁴³ which classifies the situations based on sectors for relative heading and bearing. One key difference in our approach is that we distinguish between two different classes of overtaking situations. In an *overtaking* situation, ownship is overtaking an obstacle, which is deemed to exist when the obstacle is ahead of ownship. In an *overtaken* situation, an obstacle overtakes ownship, which is deemed to exist when the obstacle is aft of ownship. This separation is necessary when designing reactive requirements, because the two situations have different required behaviors by COLREG.

When selecting the COLREG situation, it is of great importance to have persistence throughout an encounter. For instance, using the rules of Tam and Bucknall alone, a head-on situation could change into a give way situation due to the course change of the avoidance maneuver. We propose to avoid this problem by using a finite-state machine (FSM), as shown in Figure 8. The FSM has seven states, and the transitions between

them are determined by the satisfaction of complementary STL formulas. Five of the states correspond to the five COLREG situations described above. The FSM starts in a *no conflict* state. It exits this state if an obstacle is on collision course, and the collision is sufficiently close in time. This is represented by the STL formula

$$\varphi_{NC}^C = d_{CPA} \leq d_{CPA, min} \wedge t_{CPA} \leq t_{CPA, min} \quad (28)$$

After exiting the no conflict state, the FSM enters an intermediate *inbound* state. From here the type of the encounter is determined, where it will transition to one of the five COLREG situations or back to the no conflict state. Each COLREG situation has a *trigger* condition (superscript T) which corresponds to the sectors for relative heading and bearing outlined above, and a *cease* condition (superscript C). The FSM will remain in the same state until the cease condition is satisfied and thus achieves persistent selection of the COLREG situation throughout an encounter. The cease condition is equal for all situations, and is represented by the t_{CPA} being negative:

$$\varphi_{OT}^C = \varphi_{OV}^C = \varphi_{GW}^C = \varphi_{SO}^C = \varphi_{HO}^C = t_{CPA} \leq 0 \quad (29)$$

The interpretation of this is that a negative t_{CPA} means that the closest point of approach is in the past, and hence the ships have passed each other. The FSM will then transition to a *No Conflict* state.

Safety distance requirement. The safety distance requirement is perhaps the most important requirement. There exist different approaches to define the safe distance, depending on, for instance, the speed, relative bearing and size of the vessels involved.^{41,44} For simplicity, the safety distance requirement only specifies a limit on the minimum separation between vessels in this case study. The separation is the Euclidean norm of the relative position vector $d = \|\mathbf{r}_i\|$. The STL formula for the safety distance requirement is

$$\varphi_{safety} = \square-(d_i \leq d_{min}) \quad (30)$$

Mission requirement. It is also important to verify that the system completes the task it was set out to do. This can, for instance, uncover deadlocks where the system is safe and COLREG compliant but useless. Lack of mission compliance can also create dangerous situations by bringing the system outside its operational design domain (ODD). For this case study, a simple mission requirement is used, which states that the vessel should not deviate significantly from its pre-planned path. The deviation from the path is captured from the cross-track error. A piece wise linear path is assumed, which gives the following expression for the cross-track error⁴⁵

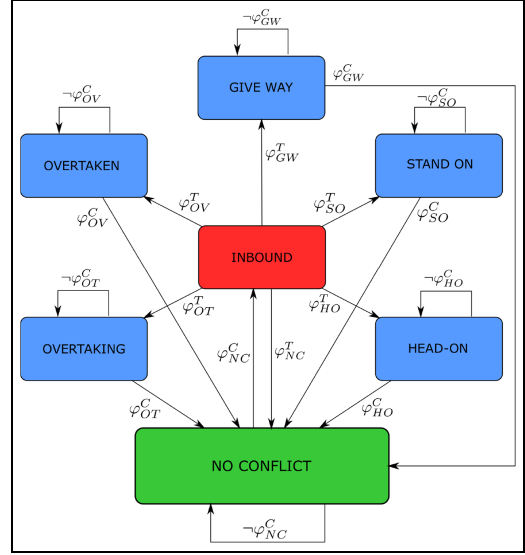


Figure 8. Finite-state machine for persistent selection of COLREG situation. Transitions are defined by complementary STL formulas. The formulas can be divided into trigger conditions (superscript T) and cease conditions (superscript C).

$$e(t) = -[N(t) - N_{wp}^k] \sin(\alpha_k) + [E(t) - E_{wp}^k] \cos(\alpha_k) \quad (31)$$

where N_{wp}^k and E_{wp}^k are the north and east position of waypoint k , and $\alpha_k = \text{atan2}(E_{wp}^{k+1} - E_{wp}^k, N_{wp}^{k+1} - N_{wp}^k)$ is the angle between path segment k and the north axis.

The STL mission requirement states that the cross-track error should always be lower than the threshold e_{max} , which enforces the vessel not to deviate too far from the preplanned path.

$$\varphi_{mission} = \square(|e| \leq e_{max}) \quad (32)$$

where $|e|$ denotes the absolute value of e .

Case 1: Obstacle on direct collision course with varying course

The first test case is a situation where ownship is traveling in a straight line, and encounters an obstacle on a direct collision course. This case has only one parameter, θ , describing the relative course of the obstacle. This is illustrated in Figure 9. The range for θ is set to $[-160^\circ, 160^\circ]$. Courses in $[-180^\circ, -160^\circ] \cup [160^\circ, 180^\circ]$ are not included as they would cause the obstacle to start unreasonably close to ownship. The hyper parameters used in the automatic testing method are given in Table 4.

Figure 10 shows the results from running the automatic testing method on Case 1 against the safety

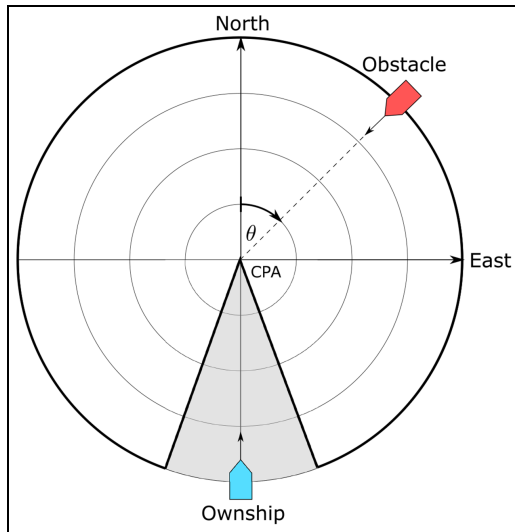


Figure 9. Definition and parameterization of Case I. The figure depicts an obstacle on direct collision course with varying course given by the parameter $\theta \in [-160^\circ, 160^\circ]$.

Table 4. Hyper parameters used in the case study.

Parameter	Case 1	Case 2
n_{conf}	3	3
κ	2	2
σ_z	0.02	0.02
σ	0.5	0.5
l	10°	$[6^\circ, 2m/s]$

distance requirement. To validate the method, 321 tightly spaced simulations were also run to get a ground truth robustness curve for comparison. The results show that this case is verified in 27 simulations with a minimum robustness of 0.6. The estimated robustness curve from the GP predicts the ground truth robustness well, and is almost always on the conservative side when there is a deviation. This is as expected, as the GP estimate will be pulled toward the prior assumption of zero robustness in regions with sparse observation. A sharp jump can be observed at $\theta = 75^\circ$. The jump corresponds to a decision boundary where the CA system transitions from maneuvering aft of the obstacle to maneuvering in front of the obstacle, which results in a sudden increase in the safety margins. Such decision boundaries are common in autonomous navigation systems and it is therefore important that the GP is able to react properly to them.

Figure 11 shows the results for the mission requirement. This run resulted in a falsification after 10 simulations, as a case with robustness less than zero was observed at $\theta = 130^\circ$. By going back and rerunning the

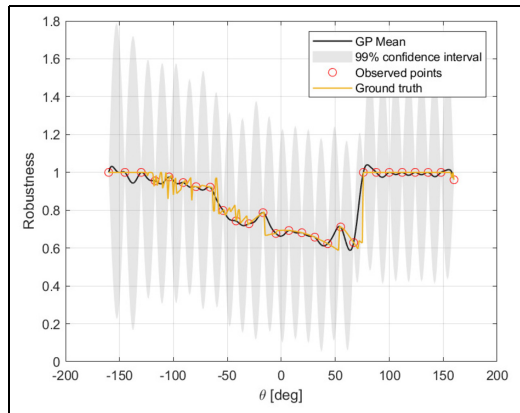


Figure 10. Robustness of the safety distance requirement in Case I. The figure shows that the requirement is verified as the lower uncertainty bound is above zero for the entire parameter space.

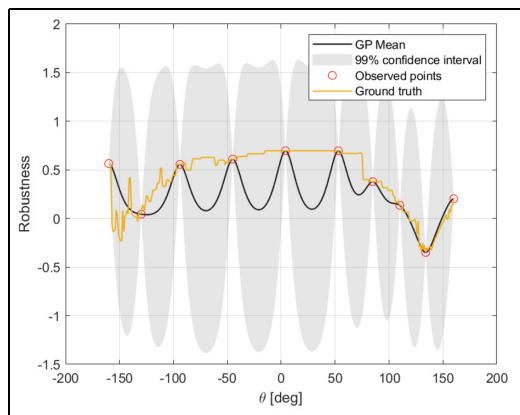


Figure 11. Robustness of the mission requirement in Case I. The figure shows that the requirement is falsified after 10 simulations as a parameter setting with robustness less than zero is observed at $\theta = 130^\circ$.

simulation with the falsified parameter setting, the falsifying behavior can be uncovered. The falsifying behavior is visualized in Figure 12. This clearly shows that the autonomous navigation system on ownship enters a deadlocked state, where it is intercepted by the obstacle and is not able to return to its planned trajectory. This highlights the importance of completeness in the requirements. Although this case passed the safety distance requirement, it still had unwanted and potentially dangerous behaviors.

Finally, Figure 13 shows the results from running the automatic testing method against the COLREG requirement. This resulted in a verification after 27 simulations. The results show good robustness over the

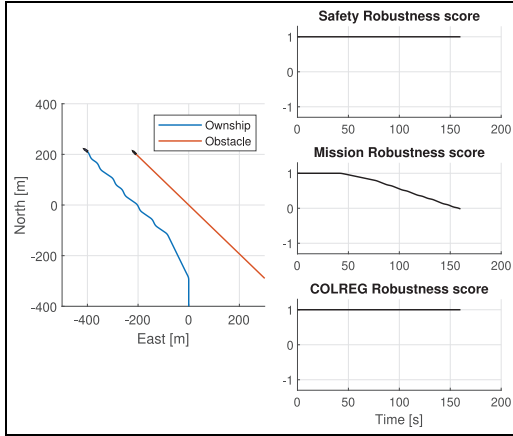


Figure 12. Falsifying behavior for the mission requirement in Case 1. The figure shows that the autonomous navigation system on ownship enters a deadlocked state, where it is intercepted by the obstacle and is not able to return to its planned trajectory.

entire parameter space. There is a step down in robustness in the interval $\theta = [22.5^\circ, 75^\circ]$. The step at 22.5° is due to a change in COLREG evaluation from *Head-on* to *Give way* and the step at 75° is again due to a decision boundary of the CA system, where it transitions from maneuvering aft of the obstacle to maneuvering in front of the obstacle.

Case 2: Head-on encounter with avoidance maneuver by obstacle

The second case is a head-on situation where the obstacle performs a predefined avoidance maneuver. This aims to test the CA systems ability to handle dynamic maneuvers by the obstacle. The case has two parameters, as illustrated in Figure 14. They are the angle of the maneuver, $\theta \in [-60^\circ, 60^\circ]$ and the speed of the obstacle $U \in [0, 20]m/s$. As the parameter space is now two-dimensional, there will be a robustness surface instead of a robustness curve as in Case 1.

We begin again by testing against the safety distance requirement. The results are shown in Figure 15. The automatic testing method resulted in a falsification after 273 simulations, as a case with robustness of -0.38 is observed at $U = 18m/s$ and $\theta = -24^\circ$. We also note that Figure 15 illustrates the adaptivity in the test case selection well, as the simulations are much denser in areas of low robustness.

The identified safety violation appears to lie on a very sharp spike of low robustness. To verify this, and assess the overall prediction of the GP, a reference robustness surface was generated by running 2501 simulations in a 61×41 grid over the parameter space. This corresponds to steps of $\theta = 2^\circ$ and $U = 0.5m/s$. The reference robustness surface is shown in Figure 16. The reference surface shows a good overall match with

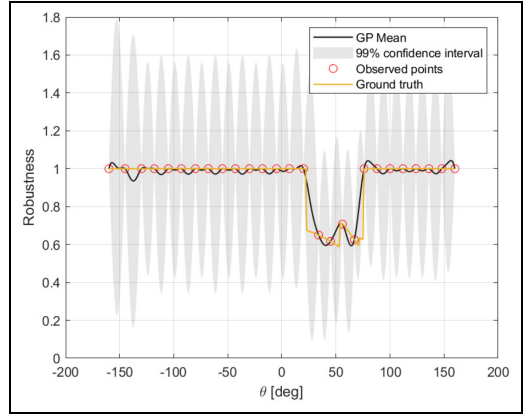


Figure 13. Robustness of the COLREG requirement in Case 1. The figure shows that the requirement is verified after 27 simulations.

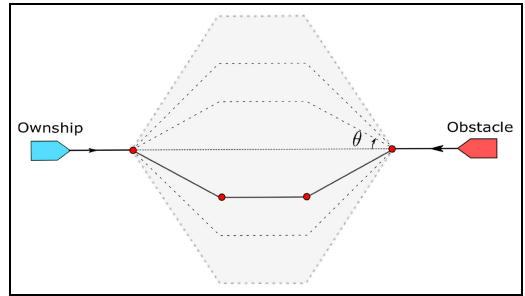


Figure 14. Illustration of Case 2. This case has two parameters, the angle of the avoidance maneuver, $\theta \in [-60^\circ, 60^\circ]$ and the speed of the obstacle $U \in [0, 20]m/s$.

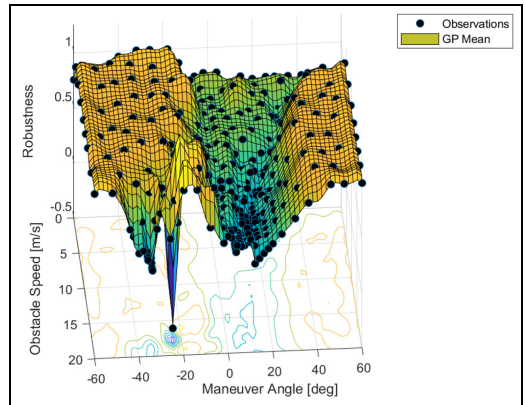


Figure 15. Inferred robustness surface and observations from running the automatic testing method against the safety distance requirement. The results show a falsification after 273 simulations.

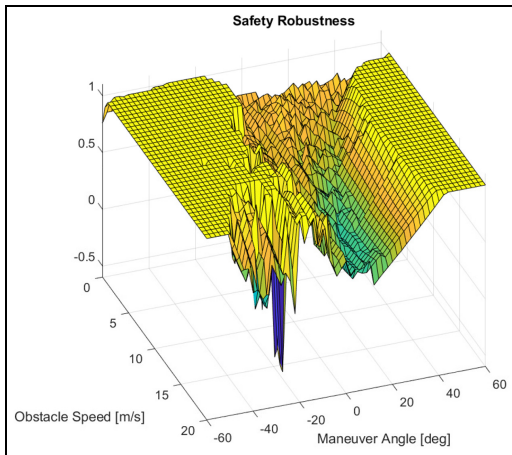


Figure 16. Ground truth robustness surface for the safety distance requirement, obtained by running 2501 simulations on a 61×41 grid over the parameter space. The surface shows a very narrow spike with low robustness.

the predicted surface by the GP and shows that indeed there is a very narrow spike of low robustness. In fact, of the 2501 simulations, only five resulted in a safety violation. This is clearly a very difficult or time consuming safety violation to detect by manual or brute force approaches. Furthermore, it illustrates that even for such a simple case with only one obstacle in open water and perfect situational awareness, safety violations that are difficult to anticipate and detect can emerge.

To examine the robustness landscape around the safety violation, 2911 simulations were ran in a dense 71×41 grid in the range $\theta = [-26^\circ, -22^\circ]$ and $U = [13, 20]m/s$, the resulting robustness surface is shown in Figure 17. This shows a sharp and narrow cleft. The vertical walls of the cleft indicate that the safety violation occurs in a region between two decision boundaries. In these boundaries, the CA system goes from full robustness to a safety violation in an arbitrarily small change in the case parameters. This type of falsification is particularly challenging to detect, because there is no gradient information in the robustness surface to guide the adaptive search toward the safety violation.

As before, we replay the simulation with the falsifying behavior to examine the cause of the safety violation. This can be very useful input for debugging and fixing the control software. The course of events is illustrated by the time-lapse in Figure 18. Ownship (blue trace) first turns starboard to do a port-port passing in accordance with COLREG Rule 14 for head-on situations. As the obstacle breaks COLREGS and turns port, ownship turns more steeply starboard to avoid a collision while still being COLREG compliant. Finally, ownship determines that a collision cannot be avoided

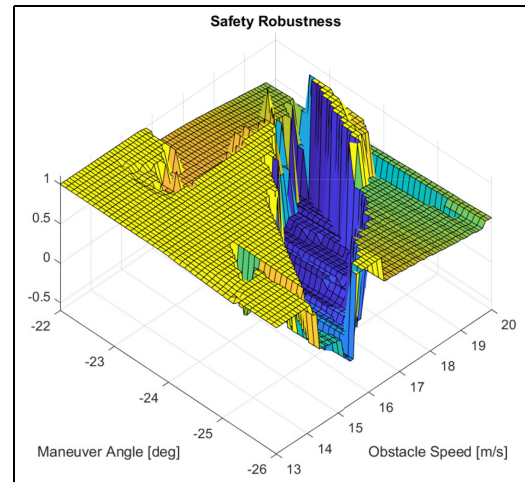


Figure 17. A closer look at the narrow spike with a safety violation, obtained by running 2911 simulations on a 71×41 grid in the range $\theta = [-26^\circ, -22^\circ]$ and $U = [13, 20]m/s$.

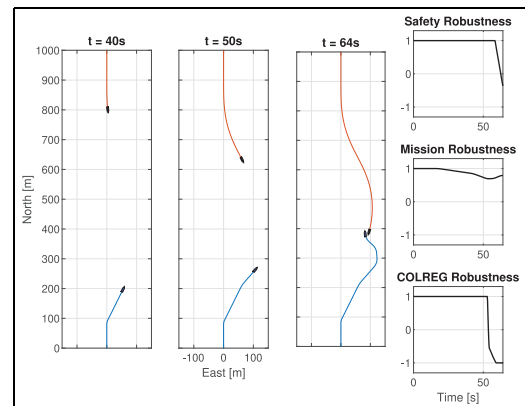


Figure 18. Time-lapse which shows the falsifying behavior identified by the automatic testing method. Ownship is shown with a blue trace, and the obstacle with an orange trace. The online STL robustness is plotted against time in the right column, which shows that both the safety distance requirement and the COLREG requirement are violated at the end of the simulation.

by a port-port passing and decides to break Rule 14 and turn steeply port. At the same time, the obstacle turns starboard and a collision occurs.

To better understand the safety violation and see what the decision boundaries represent, simulations from both sides of the cleft were also replayed. The selected cases all had $U = 16m/s$ and $\theta = [-26^\circ, -25^\circ, -24^\circ]$. This corresponds to points on the right, in the middle and to the left of the cleft in Figure 17. The simulations showed that for $\theta = -24^\circ$, the CA system decided to be compliant to Rule 14

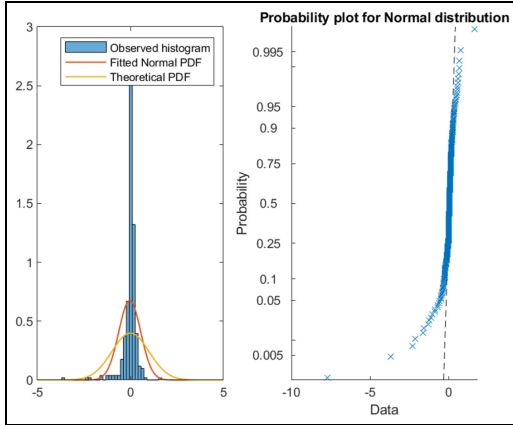


Figure 19. Statistical validation of the safety verification in Case 1. The left plot shows the histogram of the observed normalized errors together with the fitted and theoretical normal distributions. To the right, a normal probability plot is shown for the fitted distribution.

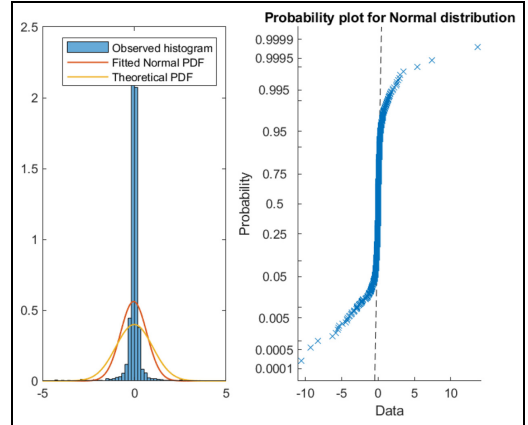


Figure 20. Statistical validation of the mission verification in Case 2. The left plot shows the histogram of the observed normalized errors together with the fitted and theoretical normal distributions. To the right, a normal probability plot is shown for the fitted distribution.

throughout the encounter. For $\theta = -26^\circ$, on the other hand, it decided early to break Rule 14 and turn port. Both of these decisions resulted in large safety margins. However, in the very narrow region in between, the CA system is indecisive and switches from the first strategy to the second at the critical moment, which results in a collision.

For brevity, the results when testing against the COLREG and mission requirements are not presented in detail, but the code and data needed to generate these results are available in an open-source online repository.⁴⁰ The key results are that testing against the mission requirement led to a verification in 267 simulations with a minimum observed robustness of 0.29. Not surprisingly, testing against the COLREG requirement resulted in a falsification after five simulations. This is not necessarily a problem, as there exists situations where it is necessary to break the explicit COLREG rules in order to navigate safely. It may nevertheless be instructive to calculate a robustness surface to get an overview of which situations the CA system decides not to be COLREG compliant and to map possible decision boundaries.

Statistical validation

Next, we apply the proposed validation method to the results from the case study. Since we are only interested in validating runs which resulted in a verification, the safety distance requirement from Case 1 and the mission requirement from Case 2 are chosen.

The reference robustness function for the safety distance requirement in Case 1, shown in Figure 10 contains 321 observations, hence we have a sample of 321

normalized errors. The distribution and normal probability plot for the normalized errors are given in Figure 19. The mean on the normalized errors is -0.036 , which shows that the distribution is close to centered about zero. The standard deviation is 0.59, which is less than the theoretical value of 1. This indicates that the GP is conservative in its confidence measure. The normal probability plot shows an S-shape, which is characteristic for distributions with light tails. This means that many of the observations are centered around zero, but there are more extreme observations than what would be expected from a normal distribution. This is backed by the histogram of the normalized errors. Further investigations showed that these extreme observations are around the discontinuity at $\theta = 75^\circ$, which is expected as the robustness function clearly disagrees with the smoothness property of the covariance function.

The reference robustness function for the mission verification in Case 2 contains 2501 observations, giving 2501 samples of the normalized error. The distribution and normal probability plot for the normalized errors are given in Figure 20. The mean on the normalized errors is -0.032 , which again shows that the distribution is close to centered about zero. The standard deviation is 0.71, again less than the theoretical value of 1. The characteristics of the observed distribution are close to those of the radial case, where again the extreme values reside around discontinuities in the robustness surface. The results from this validation indicate the GP model for the most part fits well with the observations. The extreme values indicate discontinuities which may require further investigation by the verifier.

Discussion

The results from the case study indicate that the proposed methodology has promise as a tool in the verification process for autonomous ships. Some aspects regarding its use in practice and the interpretation of the results are discussed next.

Measurement uncertainty in the Gaussian process

As shown in (13), the uncertainty when making observations can be modeled as independent Gaussian random variables. In our case, the observations are deterministic simulations which have no uncertainty associated with them. It has, however, shown to be useful to add a small observation uncertainty to relax the model fit. As shown in (19), the inference step involves the inversion of the matrix $[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I}]$. If the distance between two observations is small, compared to the length scale of the covariance function, $\mathbf{K}(\mathbf{X}, \mathbf{X})$ will have two rows which are very similar, and it will therefore be close to singular. Adding some measurement noise adds a positive number to the diagonal and therefore increases the condition number of the matrix to be inverted. This also makes intuitive sense, as the measurement noise implies that the GP mean does not need to exactly match each observation. The low condition number of $[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I}]$ manifests itself as oscillations in the GP mean surrounding sharp changes in the robustness. Some tendencies for this can be seen at the discontinuity in Figure 10 and at the falsifying downward spike in Figure 15. This has been greatly mitigated by adding an artificial measurement uncertainty.

There also exist other possible approaches to make it easier to fit a GP to the robustness surface, which instead of relaxing the GP fit by adding artificial uncertainty in the measurement, try to regularize the robustness surface. One approach is to use a smooth robustness operator⁴⁶ to obtain a smoother surface. Another approach is to use interface aware STL,⁴⁷ which distinguishes between input and output signals, and only calculates robustness on the output signals. This can remove some discontinuous jumps in the robustness surface. Both of these directions stand out as interesting candidates for future work.

Choice of hyper parameters

The choice of hyper parameters is an important part of using the proposed testing method. Since there are several of them, this gives large flexibility to tweak the verification results to the needs of the user. Therefore, it is important to have good a process for objectively selecting them. We share some experiences and insight for this next.

The hyper parameters are listed in Table 4. n_{conf} can be freely chosen up front by the user, depending on the desired confidence level. This can for instance be based on a risk analysis which determines the consequence of

breaking a certain requirement for this test case. κ determines the trade-off between exploration and exploitation when selecting the next simulation to run. This does not have a large effect on the final prediction of the robustness surface, but it may affect how many simulations are needed to obtain it. Generally, small values are favorable for fast falsification, as the search is guided toward areas with low robustness, whereas large values are favorable for fast verification, as the search is guided toward unexplored areas with high uncertainty. We found that $\kappa = 2$ offered a good trade-off in all runs. This corresponds to selecting the point which has the lowest 95% confidence interval. As discussed above, the measurement noise, σ_ϵ , should be close to zero, but a small positive number is preferable to stabilize the inference. We found that $\sigma_\epsilon = 0.02$ gave a stable inference without having a noteworthy effect on the uncertainty level.

Finally comes the choice of kernel function and its parameters. We used the ARD Matérn 5/2 Kernel,⁴⁸ which is a generalization of the simple Squared Exponential introduced in (12), with separate length scales for each case parameter. This kernel offered better handling of sharp edges, where the Squared Exponential tended to introduce oscillations in the robustness surface. The Matérn kernel has the same parameters as the Squared Exponential, the variance σ , and the length scales l for each case parameter.

The σ hyper parameter is a scaling factor which determines the scale of the assumed variations in the robustness. We chose it to give a reasonable prior distribution based on the fact that all robustness values are in the interval $[-1, 1]$. The prior distribution is zero-mean with variance of σ^2 over the entire parameter space. By choosing $\sigma = 0.5$ we get a prior which assumes that 95% of the robustness values are in the interval $[-1, 1]$. This choice gave good results for all runs.

The length scales l describes how smooth the robustness function is. Our experience is that this is the hyper parameter which has the greatest effect on the verification result and which requires the most attention when creating the GP model. This parameter determines how quickly the uncertainty increases away from observations, and therefore it essentially determines how many simulations are needed to cover the parameter space with the desired uncertainty. If the length scales are chosen too large, the verification can miss sharp spikes, such as the one in Figure 15. A practical approach to this is to first select the length scales based on how many simulations are feasible to run. We found that using a ratio between the width of the parameter range and the length scale of around 20 offered a good starting point, resulting in about 30 simulations for the one-parameter runs and 300 simulations for the two-parameter runs. Then, the results can be validated using the proposed method described above. If the length scales are chosen too large, this should be reflected in the distribution of the normalized errors by

a standard deviation greater than 1 and possibly a non-zero mean. The length scales should then be decreased.

Usage in the approval process for autonomous vessels

Finally, we discuss the context in which this method can be used in the approval process for autonomous vessels. Both the classification society DNV⁴⁹ and Norwegian Maritime Directorate⁵⁰ have proposed approval processes which by large contain the same main steps. Starting with establishing the Concept of Operations (CONOPS) which gives input to a Preliminary Hazard Analysis (PHA/HAZID) at an early stage, and later a detailed risk analysis. The output of the risk analysis will be a set of identified risks. To get an approval, evidence needs to be produced that all of the identified risks are adequately mitigated. This evidence can come in many forms, such as documentation, expert judgment, analytical results, and physical tests. For autonomous vessels it has, however, become evident that the complexity makes it necessary to do extensive simulation-based testing to verify some claims. The testing method proposed in this paper can produce evidence for some the safety claims that are difficult to prove otherwise. This can for instance be done as shown in this case study, where some traffic situations which are considered important or critical are selected and tested against a predefined set of requirements.

Another possible use is to link it more directly to the risk analysis. Systems Theoretic Process Analysis (STPA) has been proposed as a risk analysis tool for autonomous vessels.⁵¹ An outcome of an STPA analysis is a set of loss scenarios with corresponding safety constraints for preventing such losses. The safety constraints could be expressed as STL requirements and the loss scenarios could be parameterized as is done in this paper, enabling automatic verification by the proposed testing methodology. How to build trust in autonomous vessels is a challenging topic and an active area of research. How to integrate the proposed methodology in an approval process and in a design process is an important area of future research.

Conclusions

We have developed a methodology for automatic simulation-based testing of control systems for autonomous vessels. The work was motivated by the need for increased test coverage and formality in the verification efforts for autonomous vessels. It aimed to achieve this by formulating requirements in the formal logic STL, which enabled automatic evaluation of simulations against requirements using the STL robustness metric. Furthermore, the work used a GP model to estimate the robustness score and uncertainty level for untested cases. The GP model was used to automatically guide

the case selection toward cases with low robustness or high uncertainty. The main contribution of our work was an automatic testing method which incrementally runs new simulations until the entire parameter space of the case is covered to the desired uncertainty level, or until a case which falsifies the requirement is identified. The methodology was demonstrated through a case study, where the test object was a CA system for a small high-speed vessel. Requirements for safety distance, mission compliance and COLREG compliance were developed. The results showed promise, by both achieving verification in feasible time and identifying falsifying behaviors which would be difficult to detect manually or using brute-force methods. An additional contribution of this work was a formalization of COLREG using temporal logic, which appears to be an interesting direction for future work.

Acknowledgements

We wish to thank Bjørn-Olav Holtung Eriksen for providing an implementation and simulator for the BC-MPC collision avoidance algorithm for use as a test subject in the case study. We sincerely thank the anonymous reviewers for their constructive feedback which has improved the quality and clarity of the paper.

Declaration of conflicting interests


The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the Research Council of Norway through the Centers of Excellence funding scheme, project number 223254 – NTNU AMOS, and the KPN ORCAS project, project number 280655.

ORCID iDs

Tobias Rye Torben  <https://orcid.org/0000-0003-2697-8579>

Ingrid B Utne  <https://orcid.org/0000-0002-0952-3970>

References

1. NFAS. Projects, https://nfas.autonomous-ship.org/resources_page/projects-page/ (2020, accessed 15 March 2021).
2. Koopman P, Ferrell U, Fratrick F, et al. A safety standard approach for fully autonomous vehicles. In: Romanovsky A, Troubitsyna E, Gashi I, Schoitsch E and Bitsch F (eds) *Computer safety, reliability, and security*. Turku, Finland: Springer, 2019, pp.326–332.
3. Pedersen TA, Glomsrud JA, Ruud EL, et al. Towards simulation-based verification of autonomous navigation systems. *Saf Sci* 2020; 129: 104799.

4. Skjetne R and Egeland O. Hardware-in-the-loop testing of marine control system. *Model Identification Control* 2006; 27(4): 239–258.
5. Johansen TA, Sørensen AJ, Nordahl OJ, et al. Experiences from hardware-in-the-loop (HIL) testing of dynamic positioning and power management systems. In: *Second international conference on technology & operation of offshore support vessels*, Rina Imarest, 2007, pp.41–50.
6. Smogeli OM. Experiences from five years of DP software testing. In: *Dynamic positioning conference*, Houston, TX: MTS, 2010, pp.1–12.
7. Smogeli O and Skogdalen JE. Third party HIL testing of safety critical control system software on ships and rigs. In: *Offshore technology conference*, Houston, TX: One Petro, 2011, pp.839–845.
8. Sørensen AJ and Ludvigsen M. Underwater technology platforms. In: Carlton, Jukes J, Choo P, Sang Y (eds) *Encyclopedia of maritime and offshore engineering*. Hoboken, NJ: Wiley, 2018, pp.1–11.
9. Donzé A. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In: *Computer aided verification*. Berlin: Springer, 2010, pp.167–170.
10. Hoxha B, Bach H, Abbas H, et al. . Towards formal specification visualization for testing and monitoring of cyber-physical systems. In: *International workshop on design and implementation of formal tools and systems*, Edinburgh: ASU, 2014, pp.1–10.
11. Dreossi T, Dang T, Donz'e A, et al. Efficient guiding strategies for testing of temporal properties of hybrid systems. In: Havelund K, Holzmann G, Joshi R (eds) *NASA formal methods*, Pasadena, CA. Heidelberg: Springer, 2015, pp.127–142.
12. Tuncali CE, Fainekos G, Prokhorov D, et al. Requirements-Driven test generation for autonomous vehicles with machine learning components. *IEEE Trans Intell Vehicles* 2020; 5(2): 265–280.
13. Bartocci E, Bortolussi L and Sanguinetti G. Data-driven statistical learning of temporal logic properties. *Lect Notes Comput Sci* 2014; 8711(600708): 23–37.
14. Woerner K. *Multi-contact protocol-constrained collision avoidance for autonomous marine vehicles*. PhD Thesis, Massachusetts Institute of Technology, 2016.
15. Stankiewicz PG and Mullins GE. Improving evaluation methodology for autonomous surface vessel COLREGS compliance. In: *OCEANS Marseille*, Marseille, France, 2019, pp.1–7.
16. Lee R, Mengshoel OJ, Saksena A, et al. Adaptive stress testing: finding likely failure events with reinforcement learning. *J Artif Intell Res* 2020; 69: 1165–1201.
17. Luckcuck M, Farrell M, Dennis LA, et al. Formal specification and verification of autonomous robotic systems: a survey. *ACM Comput Surv* 2019; 52(5): 1–41.
18. Shokri-Manninen F, Vain J and Wald'en M. Formal verification of COLREG-based navigation of maritime autonomous systems. In: de Boer F, Cerone A (eds) *Software engineering and formal methods*. Amsterdam, The Netherlands: Springer, 2020, pp.41–59.
19. Park J and Kim J. Autonomous docking of an unmanned surface vehicle based on reachability analysis. In: *International conference on control, automation and systems*, Busan, South Korea, 2020, pp.962–966. New York: IEEE.
20. Foster S, Gleirscher M and Calinescu R. Towards deductive verification of control algorithms for autonomous marine vehicles. In: *Proceedings of the IEEE international conference on engineering of complex computer systems*, Singapore, 2020, pp.113–118.
21. Kapinski J, Deshmukh JV, Jin X, et al. Simulation-based approaches for verification of embedded systems. *IEEE Control Syst Mag* 2016; 36: 45–64.
22. Clarke EM Jr, Grumberg O and Kroening D. *Model checking*. Cambridge: MIT Press, 2018.
23. Asarin E, Dang T, Frehse G, et al. Recent progress in continuous and hybrid reachability analysis. In: *Proceedings of the 2006 IEEE conference on computer aided control systems design, CACSD*, Munich, Germany, pp.1582–1587. New York: IEEE.
24. Duffy DA. *Principles of automated theorem proving*. Chichester, NY: John Wiley & Sons, 1991.
25. Nuzzo P, Ozay N, Finn JB, et al. A contract-based methodology for aircraft electric power system design. *IEEE Access* 2014; 2: 1–25.
26. Khalil HK. *Nonlinear systems*. 2nd ed. Hoboken, NJ: Prentice-Hall, 2002.
27. Chen CT. *Linear system theory and design*. 3rd ed. New York: Oxford University Press, 1999.
28. Goebel R, Sanfelice RG and Teel AR. Hybrid dynamical systems. *IEEE Control Syst Mag* 2009; 29(2): 28–93.
29. Pnueli A. The temporal logic of programs. In: *Proceedings of the annual IEEE symposium on foundations of computer science*, Providence, RI, 1977, pp.46–57. New York: IEEE.
30. Alur R and Henzinger TA. Real-Time logics: complexity and expressiveness. *Inf Comput* 1993; 104: 35–77.
31. Maler O and Nickovic D. Monitoring temporal properties of continuous signals. In: Lakhnech Y, Yovine S (eds) *Formal techniques, modelling and analysis of timed and fault-tolerant systems*. Grenoble, France: Springer, 2004, pp.152–166.
32. Deshmukh JV, Donzé A, Ghosh S, et al. Robust online monitoring of signal temporal logic. *Formal Methods Syst Des* 2017; 51(1): 5–30.
33. Hekmatnejad M, Yaghoubi S, Dokhanchi A, et al. Encoding and monitoring responsibility sensitive safety rules for automated vehicles in signal temporal logic. In: *17th ACM-IEEE international conference on formal methods and models for system design*, La Jolla, CA 2019, pp.1–11. New York: ACM.
34. Wongpiromsarn T, Topcu U and Murray RM. Receding horizon temporal logic planning for dynamical systems. In: *Proceedings of the IEEE conference on decision and control*, Shanghai, China, 2009, pp.5997–6004.
35. Raman V, Donz'e A, Sadigh D, et al. . Reactive synthesis from signal temporal logic specifications. In: *Proceedings of the 18th international conference on hybrid systems: Computation and control*, Seattle, Washington, 2015, pp.239–248. New York: ACM.
36. Fainekos GE and Pappas GJ. Robustness of temporal logic specifications for continuous-time signals. *Theor Comput Sci* 2009; 410(42): 4262–4291.
37. Varnai P and Dimarogonas DV. On robustness metrics for learning STL tasks. In: *Proceedings of the American control conference*, Denver, CO 2020, pp.5394–5399.
38. Rasmussen CE and Williams CKI. *Gaussian processes for machine learning*. Cambridge: MIT Press, 2006.
39. McKay MD, Beckman RJ and Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 2000; 42(1): 55–61.

40. Torben TR (ed.). AutoSimTest, <https://github.com/tobiastorben/AutoSimTest> (2021, accessed 14 September 2021).
41. Eriksen BH, Breivik M, Wilthil EF, et al. The branching-course model predictive control algorithm for maritime collision avoidance. *J Field Robot* 2019; 36(7): 1222–1249.
42. IMO. COLREGS-international regulations for preventing collisions at sea. In: *Convention on the international regulations for preventing collisions at sea*, 1972, pp.1–74. London: IMO.
43. Tam C and Bucknall R. Collision risk assessment for ships. *J Mar Sci Technol* 2010; 15(3): 257–270.
44. Bakdi A, Glad IK, Vanem E, et al. AIS-based multiple vessel collision and grounding risk identification based on adaptive safety domain. *J Mar Sci Eng* 2020; 8(1): 5.
45. Fossen TI. *Handbook of marine craft hydrodynamics and motion control*. Hoboken, NJ: Wiley, 2011.
46. Pant YV, Abbas H and Mangharam R. Smooth operator: Control using the smooth robustness of temporal logic. In: *First annual IEEE conference on control technology and applications*, Maui, HI, 2017, pp.1235–1240. New York: IEEE.
47. Ferrère T, Nickovic D, Donzé A, et al. Interface-aware signal temporal logic. In: *Proceedings of the 2019 22nd ACM international conference on hybrid systems: computation and control*, Montreal, Quebec, Canada, 2019, pp.57–66.
48. Neal RM. *Bayesian learning for neural networks*. New York: Springer-Verlag, 1996.
49. Dnv GL. Autonomous and remotely operated ships-class guideline. *Technical report* September, DNV GL, 2018.
50. NMD. Føringer i forbindelse med bygging eller installering av automatisert funksjonalitet, med hensikt å kunne utføre ubemannet eller delvis ubemannet drift. *Technical report, NMD*, 2020.
51. Rokseth B, Haugen OI and Utne IB. Safety verification for autonomous ships. In: *MATEC web of conferences 2019; 273(ICSC-ESWC 2018)*, Esbo, Finland, p.02002.

Paper B:

Towards Contract-based Verification for Autonomous Vessels

**Tobias Rye Torben, Øyvind Smogeli, Jon Arne Glomsrud, Ingrid B.
Utne and Asgeir J. Sørensen**

Ocean Engineering
Volume 270, 15 February 2023, 113685
doi: <https://doi.org/10.1016/j.oceaneng.2023.113685>.

Towards Contract-based Verification for Autonomous Vessels

Tobias Rye Torben^{a,*}, Øyvind Smogeli^{a,b}, Jon Arne Glomsrud^c, Ingrid B. Utne^a and Asgeir J. Sørensen^a

^aCentre for Autonomous Marine Operations and Systems (AMOS), Norwegian University of Science and Technology (NTNU), Trondheim, Norway

^bZeabuz AS, Trondheim, Norway

^cGroup Research and Development, Det Norske Veritas (DNV), Høvik, Norway

ARTICLE INFO

Keywords:

Autonomous vessels
Contract-based design
Verification
Formal Methods
Simulation-based testing

ABSTRACT


Design and verification of autonomous vessels represent a major interdisciplinary engineering challenge due to the combination of high system complexity and the interaction with dynamic, uncertain, and unstructured environments. This paper investigates the use of contract-based methods to address both design and verification challenges of control systems for autonomous vessels. The paper first presents a formal framework for specification of components and assume-guarantee contracts using the syntax of the Z3 automated theorem prover. Then, the paper proposes a methodology for contract-based verification using the formal framework. The methodology is divided into 4 steps: (1) Hazard identification between the autonomous vessel and the operative environment in order to define the top-level component and contract, (2) stepwise refinement of the top-level component into detailed sub-components and contracts, (3) definition of test setups for simulation-based testing to verify that components meet their contract, and (4) applying a recursive procedure for contracts-based system verification. The framework and methodology are demonstrated in a case study with an autonomous passenger ferry.

1. Introduction

The control systems that govern modern ships are continuously increasing in complexity and becoming more software intensive. The recent development of autonomous vessels represents the ultimate culmination of this trend. Building on top of existing advanced maritime guidance, navigation, and control systems, autonomous vessels must additionally be capable of obtaining situational awareness and performing intelligent planning and decision making under uncertainty in the dynamic and unstructured maritime environment. The development of autonomous vessels is enabled by recent advances in control, optimization, planning, artificial intelligence, computer vision, and sensor fusion, in addition to the ever-increasing computational resources available for embedded systems. However, although these technological advances allow us to implement the different functionalities necessary for autonomy, combining all of them into an integrated system, layered on top of the already complex maritime control systems, results in an unprecedented level of system complexity. In addition to the sheer complexity, verification of autonomous systems is generally very hard due to their extensive sensing and interaction with the open environment. This leads to an infinite number of unknown scenarios that an autonomous vessel may encounter in operation, and it will thus never be possible to specify the required behavior in every scenario during design (Torben, Smogeli, Utne and Sørensen, 2022b; Murray, Rødseth, Nordahl, Wengersberg, Pobitzer and Foss, 2022). To design reliable and robust autonomous vessel control systems and verify their safety and functionality, there is a clear need for new methodology both in design and verification.

Several recent works propose methodologies for verification of autonomous vessel control systems. The use of Systems-Theoretic Process Analysis (STPA) has been proposed for deriving safety requirements to verify against (Rokseth, Haugen and Utne, 2019; Chaal, Valdez Banda, Glomsrud, Basnet, Hirdaris and Kujala, 2020). The use of STPA in combination with Bayesian Belief Networks has also been proposed as a design methodology to give autonomous vessels an online risk management capability (Utne, Rokseth, Sørensen and Vinnem, 2020). Simulation-based testing is frequently proposed as a methodology to produce verification evidence for autonomous vessels (Pedersen, Glomsrud,

*Corresponding author

 tobias.torben@ntnu.no (T.R. Torben)

 <https://www.ntnu.edu/employees/tobias.torben> (T.R. Torben)

ORCID(s): 0000-0003-2697-8579 (T.R. Torben)

Ruud, Simonsen, Sandrib and Eriksen, 2020). Existing work addresses simulation-based testing of collision avoidance aspects (Woerner, Benjamin, Novitzky and Leonard, 2019; Bakdi, Glad and Vanem, 2021) and situational awareness aspects (Vasstein, 2021). Some work is also emerging on formal methods for design and verification of autonomous vessels (Shokri-Manninen, Vain and Waldén, 2020; Foster, Gleirscher and Calinescu, 2020). We strongly believe that design and verification of autonomous vessels needs to be modular in order to manage the immense complexity. The current research frontier addresses methodology for system-level risk assessment and requirement derivation as well as methodology to verify certain modules of autonomous vessels. However, we see an unresolved need for a modular methodology to go from system-level requirements and verification to detailed module requirements and verification in a coherent and coordinated way. We propose a contract-based approach as a candidate to address these needs.

Contract-based design is a modular approach to system design. The system is modularized into encapsulated design units called *components*. Each component is associated with an assume-guarantee contract which specifies what the component assumes about its environment and what behavior it can guarantee given that these assumptions hold. Contract-based design can also support stepwise refinement, where a high-level contract for a component at a high abstraction level is incrementally refined into more detailed contracts for sub-components at lower abstraction levels (Abrial, 2011). The ultimate goal of taking a contract-based design approach is to enable compositional reasoning, that is, reasoning about the correctness of a composed system based on the contracts of individual components. Compositional reasoning includes both checking that the contracts of connected components are compatible and that the composition of contracts for sub-components correctly refines the higher-level contract. By taking on a suitable mathematical formalism when specifying the contracts, it is also possible to reason about the correctness of a composition or refinement step in a formal, mathematical manner (Cimatti and Tonetta, 2012). The use of contracts for specification was originally introduced in the context of software engineering, with the *design by contract* methodology of Meyer (1992). In the same period, the foundation for compositional reasoning was also developed in the pioneering work of Clarke, Long and McMillan (1989) as a divide-and-conquer solution to the scalability issues of model checkers. Over the past two decades, significant research has attempted to apply contract-based methods to cyber-physical systems resulting in more complete frameworks combining specification, refinement, and verification (Sangiovanni-Vincentelli, Damm and Passerone, 2012; Nuzzo, Sangiovanni-Vincentelli, Bresolin, Geretti and Villa, 2015). Contract-based methods have seen application in other industries which are faced with complex and safety critical systems, such as aviation (Nuzzo, Xu, Ozay, Finn, Sangiovanni-Vincentelli, Murray, Donzè and Seshia, 2014) and automotive (Benveniste, Caillaud, Ferrari, Mangeruca, Passerone and Sofronis, 2008). In the maritime industry, however, contract-based methodologies have not yet seen any adoption. A notable exception is the recent publication of Hake, Hohl and Hahn (2021), which proposes a contract-based methodology for verifying software updates on shipboard equipment. The authors are not aware of any previous research targeting contract-based verification for autonomous vessels.

We believe that taking a contract-based design approach has the potential to address many of the challenges related to design and verification of autonomous vessel control systems. Having modularity in a design is a well-proven technique for decomposing and managing complexity. Modularity often also comes as a necessity in maritime control systems when integrating custom and commercial-off-the-shelf (COTS) components from several different vendors. Having a more structured and formalized integration of control systems and equipment from different vendors is a long-standing need in the maritime industry (Smogeli, Ludvigsen, Jamt, Vik, Nordahl, Kyllingstad, Yum and Zhang, 2020). Another challenge for complex systems in general, and autonomous systems in particular, is to derive a complete and coherent set of system requirements to ensure safe and correct behavior (Rokseth and Utne, 2019). In a contract-based setting, the contracts act as the requirements for individual components. Because components can be structured hierarchically, meaning that a component can be implemented by a set of sub-components, we can analyze the system at different abstraction levels. This may simplify the derivation of requirements, as the derivation of detailed sub-components contracts often emerges naturally as a combination of refining the higher-level contract and being compatible with other sub-component contracts. Refinement checking ensures the completeness of contracts with respect to the top-level contract. Another benefit of deriving assume-guarantee contracts is that all assumptions in the system are made explicit and may be monitored online as an additional safety function. In addition to compositional reasoning about contracts, a central step in the verification process is to show that each component complies with its contract. Due to the complexity and the types of algorithms used in autonomous vessel control systems, formal verification of contract compliance will likely be impossible with the current state-of-the-art. Therefore, we propose simulation-based testing as an alternative solution for verifying contract compliance (Pedersen et al., 2020). Although exhaustive verification

Table 1

Overview of the most used symbols in this paper.

σ	Behavior
M	Component
C	Contract
A	The set of behaviors that define the assumptions of an assume-guarantee contract
G	The set of behaviors that define the guarantee of an assume-guarantee contract
φ_A	The logic formula which specifies the assumptions A of a contract
φ_G	The logic formula which specifies the guarantees G of a contract
b_C	Observer for the contract C

can not be achieved with a testing method, simulation-based testing already has a strong and successful history in the verification of maritime control systems (Smogeli and Skogdalen, 2011; Smogeli, 2015), and several methodologies exist for increasing the exhaustiveness (Torben, Glomsrud, Pedersen, Utne and Sørensen, 2022a). It will be beneficial to use a combination of several different simulators to achieve both realistic and integrated simulations and high test coverage. We believe that a contract-based framework can be used together with simulation-based testing in a mutually beneficial way. The simulation-based testing acts as a means to generate evidence of contract compliance, and the contract framework ensures that the testing efforts combine in a structured and coordinated way in order to build evidence of the overall system correctness.

The research objective of the current work is to investigate if a contract-based design approach can enable more structured and formalized verification of autonomous vessel control systems, and the main scientific contribution is a top-down methodology for contract-based verification of such systems. This paper is outlined as follows. In Section 2 we first introduce some mathematical foundations before we present a framework for defining components and specifying contracts using the Z3 theorem prover (de Moura and Bjørner, 2008). In Section 3 we present a methodology for contract-based verification using the framework of Section 2. The methodology is divided into 4 steps: (1) Hazard identification between the autonomous vessel and the operative environment in order to define the top-level component and contract, (2) stepwise refinement of the top-level component into detailed sub-components and contracts, (3) definition of test setups for simulation-based testing to verify that components meet their contract, and (4) applying a recursive procedure for contracts-based system verification. In Section 4, we demonstrate the use of the contract-based framework and methodology in a case study with an autonomous passenger ferry. In Section 5 we discuss some challenges of the methodology and possible approaches to address these. Concluding remarks and suggestions for future work are given in Section 6.

2. Contract framework for system verification

In this section, we introduce the contract framework used in this work. We begin by presenting the mathematical foundations, where components and contracts are abstractly defined in terms of sets of behaviors. Then we go on to introduce the Z3 theorem prover and show concretely how components and contracts can be specified using the syntax of Z3. Table 1 is included to give an overview of the most used symbols.

2.1. Preliminaries: Mathematical foundation for contract-based design

The mathematical foundation for the framework is defined using set notation. This is based on the assume-guarantee contract theory from Benveniste, Caillaud, Nickovic, Passerone, Ralet, Reinkemeier, Sangiovanni-Vincentelli, Damm, Henzinger and Larsen (2018), to which the reader is referred for further details.

Consider the variables v_1, v_2, \dots, v_n , each with domain D_1, D_2, \dots, D_n . Let $V := D_1 \times D_2 \times \dots \times D_n$. A *reaction* $s \in V$ is defined as a vector assigning values from V to each variable. A *behavior* σ is defined as a, possibly infinite, discrete-time sequence of reactions $\sigma = s_1, s_2, s_3, \dots$. We add a special symbol, \perp , to the domain of each variable, to indicate the absence of a reaction to a particular variable and define the *silent reaction* ϵ as the reaction which assigns \perp to each variable. For simplicity, we consider only synchronous behaviors here, where each variable is assigned a reaction simultaneously at discrete time steps. Extensions to consider asynchronous behaviors may be achieved using

Kahn Process Networks (Kahn, 1974), as proposed by Benveniste et al. (2018).

A *component*, M , is described abstractly in terms of the set P of behaviors the component exhibits. In practice, the set of behaviors for a component can for instance be specified in terms of a differential equation or a computer program. The *composition* of two components $M_1 \times M_2$ is defined as the intersection of their respective sets of behaviors $P_1 \cap P_2$.

A *contract* is defined as a pair of assertions $C = (A, G)$, where A are called the assumptions and G are called the guarantees. The set \mathcal{E}_C of legal environments is the collection of all components E for C , such that $E \subseteq A$. The set \mathcal{M}_C of legal implementations of C is defined by the collection of components M such that $A \times M \subseteq G$. All contracts which admit the same set of behaviors are by definition equivalent. A contract is *saturated* if $G = G \cup A^c$, where A^c is the complement of A . Contract saturation means that the set of guarantees is maximal in the sense that it contains all behaviors where the assumptions do not hold. Since the guarantees only are in force when the assumptions hold, all contracts can be transformed into an equivalent saturated contract.

Next, we define a set of contract operations. Let $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$ be two saturated contracts defined over the same set of variables. The refinement relation $C_2 \leq C_1$ is defined by

$$C_2 \leq C_1 \text{ iff } G_2 \subseteq G_1 \text{ and } A_2 \supseteq A_1.$$

If this relation is satisfied, we say that C_2 refines C_1 . Refinement is an ordering of the relative strength of contracts. Informally, a contract has to have as strong or stronger guarantees and as permissive or more permissive assumptions as another contract in order to refine it.

The *conjunction* of C_1 and C_2 , denoted $C_1 \wedge C_2$, is defined as

$$C_1 \wedge C_2 := (A_1 \cup A_2, G_1 \cap G_2).$$

Typically, conjunction is used to impose several different contracts on a component, such that the component needs to comply with each of them in order to comply with the conjunction.

The *composition* of two contracts is denoted $C_1 \otimes C_2$. Composition is defined as

$$C_1 \otimes C_2 := (G_1 \cap G_2, (A_1 \cap A_2) \cup (G_1 \cap G_2)^c)$$

Similar to the composition of components, composition of contracts can be used to construct composite contracts out of simpler ones. For instance, if a component is implemented by a set of sub-components, we may want to verify that the composition of the contracts for the sub-components refines the contract of the parent component. From the definition of composition, it can be seen that a component has to comply with the guarantees of both C_1 and C_2 . However, the assumptions of the composite contract are relaxed, as some of the assumptions may be covered by the guarantees of the other contract. It can be shown that using these definitions, the conjunction and composition operators are associative and commutative.

Finally, we define the concept of *observers*. Observers are in this context used for evaluating whether a single behavior complies with a contract or not, which is central when testing a component. For a contract $C = (A, G)$ and a behavior σ , the observer $b_C(\sigma, A, G) = \text{True}$ iff $\sigma \in G \cup A^c$. Otherwise, $b_C(\sigma, A, G) = \text{False}$.

2.2. Contract framework using the Z3 theorem prover

We continue by presenting a concrete contract framework that uses the syntax of the Z3 automated theorem prover to specify components and contracts and show how this relates to the abstract framework of Section 2.1.

2.2.1. A short intro to Z3

Z3 is an open-source automated theorem prover (ATP) developed by Microsoft Research (de Moura and Bjørner, 2008). ATPs are tools that are given a mathematical theorem as input and attempt to automatically prove or disprove it. Z3 is an ATP of the Satisfiability Modulo Theories (SMT) solver type. To understand SMT solvers, we first introduce the simpler and more well-known boolean satisfiability solvers, commonly referred to as SAT solvers. SAT solvers

attempt to decide if a boolean expression in the form of a propositional logic formula is satisfiable or not, that is, if there is a combination of true or false assignments to the boolean variable which makes the formula true. SAT solving is nondeterministic polynomial-time complete (NP-complete), and a wide range of important NP-complete problems can be rephrased as SAT problems. Significant effort has therefore been put into developing efficient heuristics for SAT solving, such that state-of-the-art SAT solvers can solve problem instances involving several hundred thousand variables.

SMT solvers generalize SAT solvers from boolean formulas to more complex formulas involving e.g. integers, real numbers, arrays, or strings. This is achieved by combining a SAT solver with a set of domain-specific theories, such as linear arithmetic. SMT solvers can be used as ATPs by checking the satisfiability of the negation of a theorem. If the negation of the theorem is found to be unsatisfiable, then the theorem is proved. If, on the other hand, a solution is found which satisfies the negation of the theorem, this will disprove the theorem, and the SMT will return the solution as a counterexample.

Z3 operates on first-order logic formulas, which in addition to the operators of propositional logic (And (\wedge), Or (\vee), Negation (\neg) and Implication (\rightarrow)), contain the universal quantifier "for all", denoted by the symbol \forall , and the existential quantifier "there exists", denoted by the symbol \exists . Z3 has bindings for several programming languages. In this paper we will use the Python bindings as the syntax to define components and contracts due to its good readability and Python being a widely known language.

2.2.2. Component model

Next we define the component model of our contract framework. The model is illustrated conceptually in Figure 1. The component interface consists of *out-ports*, which are controlled by the component, and *in-ports*, which are controlled by the environment of the component. In addition, a component can have a set of *parameters* which are constant during a simulation. The communication between components is achieved by sending *messages* on out-ports that are received on the in-ports of other components. The data structure of a message is defined by its *message type*. The message type is defined by declaring a set of variables defined by a variable name and data type. Variables can be nested to form struct-like data structures. The allowed data types are the Z3 basic data types *Real*, *Int* and *Bool*. The basic data types also have fixed-length vectorial versions, denoted *RealVector*, *IntVector* and *BoolVector*. We note that Z3 supports several other datatypes, such as strings, arrays, bit vectors and even floating point numbers, however, this is not explored further in our work. For examples of message type definitions, the reader is referred to the case study in Section 4.2.

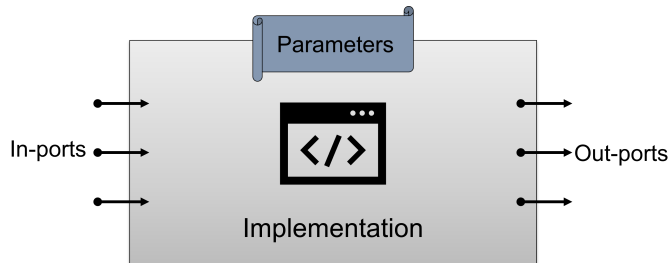


Figure 1: Conceptual view of the component model used in this work.

The *implementation* of a component synchronously reads messages from the in-ports and assigns messages to the out-ports at each discrete time step. Assignment of a *None* object is used to indicate the absence of an assignment (similar to the silent reaction in Section 2.1). We separate between two types of components: Composite components, which are implemented by composition of sub-components, and atomic components which are implemented directly, for instance as a C program or a ROS node. The fact that a component can be implemented by a set of sub-components is a key concept in our component model. This means that components can be structured hierarchically, which enables a system to be analysed at different *abstraction levels*. This is an effective approach to manage the complexity both

during the design and verification phases. The following section will show how we propose to utilize the hierarchical component structure in our verification framework. An example of a composite component is given in Figure 2.

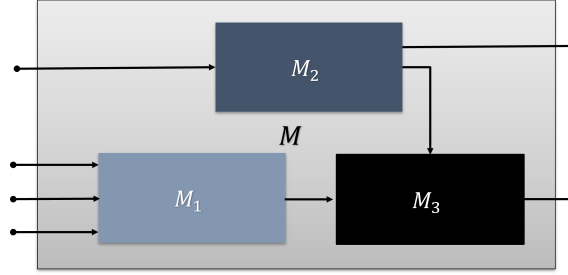


Figure 2: Example of a composite component structure with two abstraction levels. Component M is implemented by the sub-components M_1 , M_2 , and M_3 such that $M = M_1 \times M_2 \times M_3$.

To draw a comparison back to the abstract framework of Section 2.1, the variables v_1, v_2, \dots, v_n correspond to the in-ports and out-ports of the component. The domains of each variable, D_1, D_2, \dots, D_n are defined by the corresponding message type of the component. A behavior is a discrete sequence of messages assigned to each port of the behavior, and the set of behaviors for the component is implicitly defined by the implementation of the component. Composition of components is achieved by connecting the out-ports of one component with the in-ports of another component with the same message type.

2.2.3. Contract specification in Z3

As introduced in Section 2.1, contracts characterize the legal environments and legal implementations for components. In assume/guarantee contracts, a contract C is defined by a pair of assertions that specify the set of legal environment behaviors, called the assumptions A , and the set of guaranteed component behaviors given that the assumptions hold, called the guarantees G . We propose to use first-order logic formulas in the Z3 syntax to specify the sets A and G of the contract.

In our framework, a contract is linked to a specific component and therefore acts as a specification of requirements for the component. To preserve modularity, a contract may only refer to the ports of the component it is linked to. A contract consists of two first-order logic formulas φ_A and φ_G which define the assumptions and guarantees, respectively. The basic building blocks of the logic formulas are *predicates*, which are functions $\pi : V \mapsto \mathbb{B}$ mapping values of the variables v to a boolean value. An example of a predicate on the variable $x \in \mathbb{R}$ is $x \leq 10$. In the formulas φ_A and φ_G , several predicates are combined using the first-order logic operators. $x \leq 10 \wedge x \geq 0$ is an example of a formula with two predicates connected by the \wedge operator. Usually, we wish to specify several assumptions and guarantees for a component. This can easily be achieved by letting the formulas φ_A and φ_G be formulated as a conjunction of individual assumptions and guarantees.

Comparing this with the abstract definition of assume/guarantee contracts in Section 2.1, the sets A and G are now defined by the logic formulas φ_A and φ_G . We only consider specification of time-invariant properties in this work, that is, properties that must hold at each time step. Extensions to consider temporal properties may be achieved by using a temporal logic, such as Signal Temporal Logic (STL) (Maler and Nickovic, 2004). For time-invariant assumptions and guarantees, the sets A and G are simply subsets of V . The formulas (φ_A, φ_G) , and the sets (A, G) are related as follows. Each predicate π_i in the logic formulas defines a set $\mathcal{O}(\pi_i) \subseteq V$ where that predicate is true. When combining predicates into a logic formula, the set that the formula defines is defined for each logical operation. Logical conjunction, $\pi_1 \wedge \pi_2$ translates to set intersection $\mathcal{O}(\pi_1) \cap \mathcal{O}(\pi_2)$. Logical disjunction, $\pi_1 \vee \pi_2$ translates to set union $\mathcal{O}(\pi_1) \cup \mathcal{O}(\pi_2)$. Logical negation, $\neg \pi_1$ translates to set complement $\mathcal{O}(\pi_1)^c$. Translation of the logical quantifiers (\forall and \exists) is also possible, but this is not explored further in our work.

As shown in Section 2.1, a behavior σ complies with the contract if $\sigma \in G \cup A^c$. In terms of the first-order logic formulas, σ complies with the contract iff $\varphi_G(\sigma) \vee \neg \varphi_A(\sigma) = \text{True}$. This coincides the definition of logical implication:

$p \rightarrow q := q \vee \neg p$. Hence, a behavior σ satisfies the contract $C = (\varphi_G, \varphi_A)$ iff $\varphi_A(\sigma) \rightarrow \varphi_G(\sigma) = \text{True}$. Moreover, this entails that the observer for C is simply defined as $b_C(\sigma, \varphi_A, \varphi_G) = \varphi_A(\sigma) \rightarrow \varphi_G(\sigma)$. For specific examples of contracts, the reader is referred to Section 4.2 of the case study.

3. Methodology for contract-based verification of autonomous vessels

In this section, we propose a step-by-step methodology for contract-based verification of autonomous vessels using the framework introduced in Section 2. Step 1 defines the top-level level component and specifies the top-level contract in order to ensure safe interactions with the operative environment. Step 2 refines the top-level components and contracts in a series of refinement steps until we reach a sufficient level of detail. Step 3 defines the simulation-based test setup for each component. Finally, Step 4 applies a recursive algorithm that verifies contract refinement and runs simulation-based testing to verify contract compliance. The methodology is illustrated in Figure 3. We give a few illustrative examples in this section. Instead, we recommend the reader to look at the corresponding steps in the case study of Section 4 for concrete examples.

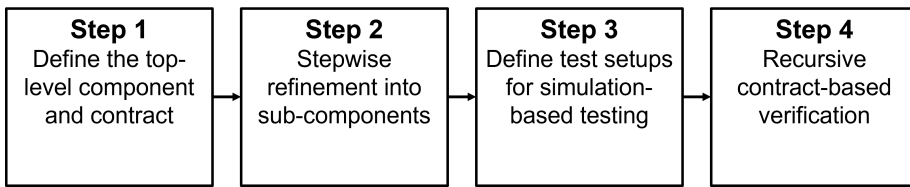


Figure 3: The 4 steps of the proposed methodology for contract-based verification.

Step 1: Define the top-level component and contract

To enable contract-based verification for an autonomous vessel, we must first define the component structure and assign contracts to the components. This can be done both during the design of a new autonomous vessel and by formulating an existing autonomous vessel design in terms of components and contracts. In Step 1, we begin at the top-level view, shown in Figure 4. This consists of two components; the autonomous vessel and its operative environment. The first step is modeling the operative environment and identifying all relevant interactions between the autonomous vessel and its operative environment. This involves creating the necessary port definitions and corresponding message types between the autonomous vessel component and the operative environment component.

Next, the top-level contract between these components must be formulated. The assumptions of the top-level contract specify which environmental conditions the autonomous vessel is designed to operate under, and thus define the operational design domain (ODD). The guarantees of the top-level contract form the system-level requirements on the behavior of the autonomous vessel.

Achieving completeness both in identifying relevant interactions with the operative environment and constraining these in the top-level contract is of paramount importance in the verification process. Achieving sufficient completeness in requirements and identifying relevant test scenarios for verification are generally major challenges in the assurance process for complex systems (Rokseth and Utne, 2019) and in particular for systems with a high level of autonomy (Rokseth et al., 2019). The STPA methodology provides a systematic top-down approach to hazard identification and generation of loss scenarios which may constitute an important foundation for increasing the completeness when identifying interactions and formulating the top-level contract. Moreover, we believe that our top-down approach simplifies the requirement identification process since the top-level contract is formulated at a high level of abstraction. More detailed requirements are either derived from the top-level contract or emerge at lower abstraction levels to constrain the interactions between sub-components. This allows us to focus on specifying the desired system-level behavior at this stage, such as keeping a minimum distance to other vessels, and leave the details regarding how these requirements are satisfied and all the ways in which they can be violated, to the more detailed requirements further down in the abstraction levels.

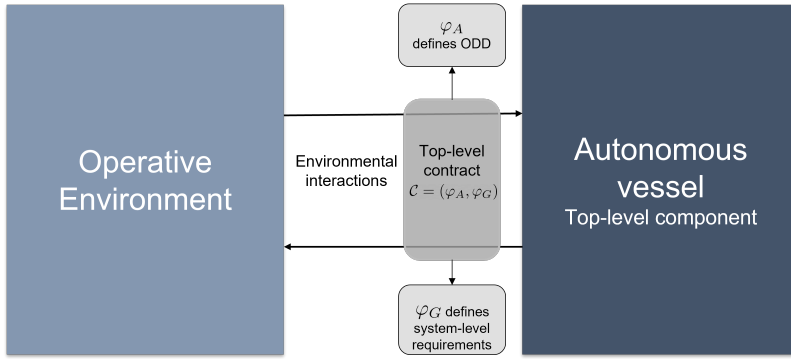


Figure 4: Step 1: Define the top-level component and contract.

Note that the assumptions of the top-level component are special. Since they are assumptions on the operative environment, they are outside of our control. Examples of environmental assumptions are assuming that the speed of other vessels is below some threshold, or that the weather conditions satisfy certain criteria. Even though we cannot control the environment to comply with these assumptions, we believe it is important to specify them. Having awareness of the assumptions of the environment is critical to designing a safe and robust system, and it allows us to monitor these assumptions and take appropriate action if they are violated.

Step 2: Stepwise refinement into sub-components

After the top-level component and contract have been established in Step 1, Step 2 involves a series of design iterations where the components and contracts are incrementally refined into a more detailed and implementation-ready form. This process takes place as a series of well-defined refinement steps. In each refinement step, a component is decomposed into a set of sub-components, as illustrated in Figure 2. Additionally, contracts must be assigned to each new sub-component. The composition of the contracts of the sub-components has to refine the contract of the component which the sub-components implement. The contracts of the sub-components also need to be compatible with each other, that is, if an out-port of component M_i with contract $C_i = (\varphi_A^i, \varphi_G^i)$ is connected to an in-port of component M_j with contract $C_j = (\varphi_A^j, \varphi_G^j)$, then the guarantees of C_i have to be strong enough to enforce the assumptions of C_j . Note that the actual checking of the correctness of the refinement is done in Step 4.

Arriving at component architecture with corresponding contracts that respect both refinement and compatibility is a design exercise that requires good engineering judgment. Architectural decisions must be made, custom components must be designed and COTS components must be selected. This is an iterative process that often involves contract negotiations. For instance, when composing a set of sub-components, one may realize that the selected sensor does not have the precision required to refine the contract of the parent component. In this case, a new design iteration is required, for instance by selecting a more precise sensor or designing a more precise controller. In fact, this is very similar to the types of decisions that engineers typically make during system design, however, the contract-based concepts of contracts, composition, and refinement give a language and framework to analyze these questions in a more structured and formalized manner (Benveniste et al., 2018).

The refinement process of Step 2 should be continued until all components are described at a level of detail that can readily be implemented in hardware or software. Moreover, all components should be decomposed until they are simple enough to be verified to the desired level of assurance. For critical components, the refinement may continue all the way to individual software functions, whereas the refinement of less critical components may be stopped at a higher level of abstraction. In some cases, the refinement stops naturally at COTS components.

Step 3: Defining test setups for simulation-based testing

Having the full system structure, in form of a component structure with corresponding contracts, Step 3 involves defining test setups for simulation-based testing of the components. We first give some background on why we propose to combine simulation-based testing and contract-based verification before we state the specific activities for Step 3.

3.1 Combining simulation-based testing and contract-based verification

Testing is a method for verifying a component where only a selected subset of the behaviors of the component is checked. This is the most common means of verification for embedded systems, because exhaustive verification is often too time-consuming or not feasible at all (Kapinski, Deshmukh, Jin, Ito and Butts, 2016). In *simulation-based* testing, the component is tested in a simulated environment. Parts of the component itself may also be simulated in order to focus testing on certain aspects. Some common component representations for simulation-based testing are Hardware-in-the-Loop (HiL), where the real component software runs on the real component hardware, Software-in-the-Loop (SiL), where the real component software runs on virtualized hardware, and Model-in-the-Loop (MiL), where a simulation model of the component is used (Torben et al., 2022a). Simulation is able to analyze the system-level behavior of highly complex systems. Autonomous vessels are characterized by high levels of complexity in their hardware and software systems, as well as in their complex interaction with the operative environment. Combined with the intrinsic challenges related to verification of autonomous functions, such as the use of machine learning components and hard-to-predict emergent behaviors, currently there exist no viable alternatives to simulation-based testing for verifying their system-level behavior. It is widely agreed upon that simulation-based testing will be a key solution for the assurance of autonomous vessels (Pedersen et al., 2020). Next, we show how existing methodologies for simulation-based testing and contract-based verification can be combined in a mutually beneficial way.

A central activity in contract-based verification is to prove that a component complies with its contract. We call this activity *contract compliance checking*. Most previous examples of contract-based verification have used formal verification techniques for contract compliance checking, such as model checking (Clarke, 1997). This is attractive, as it results in a rigorous mathematical proof of contract compliance. However, due to the overall complexity, the hybrid system dynamics, and the use of advanced control techniques such as machine learning algorithms and model predictive control (MPC), state-of-the-art formal verification techniques are not capable of formally verifying all aspects of autonomous vessel control systems. Thus, we propose simulation-based testing as an alternative approach to contract compliance checking. Instead of formally verifying that all behaviors of a component comply with the contract of the component, simulation-based testing will generate evidence of contract compliance by running an adequate number of simulations in well-chosen scenarios and evaluating the resulting behaviors against the contract using an observer for the contract. Existing methodology for simulation-based testing, which addresses scenario selection and coverage assessment can readily be used for contract compliance checking.

Hence, simulation-based testing may solve a problem for contract-based verification of complex autonomous systems. At the same time, contract-based verification addresses known challenges for simulation-based testing. High-fidelity simulation of an entire autonomous vessel and its operative environment is possible using 3D-rendering engines for exteroceptive sensor simulation and software-in-the-loop for including exact replicas of the control software in the simulations. However, such simulations are typically highly computationally expensive resulting in low simulation speed and corresponding limited test coverage. Many sub-systems can be simulated accurately using simplified models that are orders of magnitude less computationally expensive, and thus achieve high test coverage. This suggests that several different simulators should be combined in order to address both system-level behavior at a high level of integration and sub-system behavior with high test coverage. However, using several different simulators raises the question of how the testing efforts by each of these should be combined in a coordinated manner in order to build verification evidence for the autonomous vessel. We believe that contract-based verification addresses this question. The component structure of the system specifies the simulator taxonomy directly since each component will need its own simulator to perform the contract compliance checking. The contracts between the components ensure that the simulation efforts combine in a structured and coherent way towards achieving an overall assurance of the autonomous vessel. Additionally, as shown in the following, contract-based verification provides a good framework for test management.

3.2 Activities for Step 3

Step 3 of the methodology involves defining the *test setup* for each component in the system. A schematic view of a generic test setup is shown in Figure 5. We split the test setup into two parts; the simulator and the test management system. The simulator part consists of the component under test in a HiL, SiL, or MiL representation, connected with a simulation model of the environment of the component, which generates the test inputs to the component under test. The simulator is connected to the test management system over a test interface, which outputs a behavior, σ , for each simulation. The behavior is evaluated against the contract $C = (\varphi_A, \varphi_G)$ of the component using the observer $b_C(\sigma, \varphi_A, \varphi_G)$, as described in Section 2.2.3. The assumptions φ_A of the contract for the component under test should also be used to define and focus the test scenarios. The scenario selection may use the evaluation of previous simulations to adaptively select new scenarios. This can for instance be achieved by guiding the scenario selections towards regions of the scenario space with poor performance or high uncertainty, as proposed in Torben et al. (2022a).

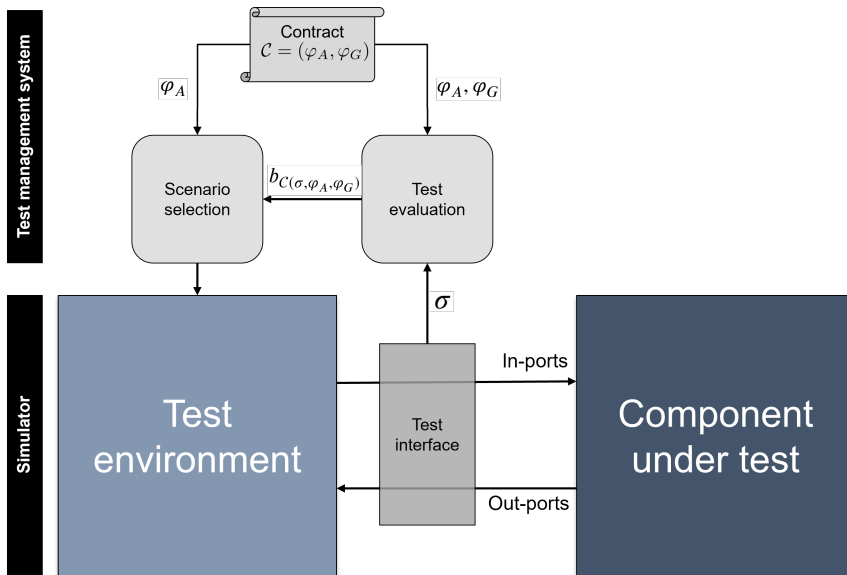


Figure 5: Step 3: Test setup for simulation-based contract compliance checking.

Step 4: Recursive contract-based verification

Having defined the component structure, assigned contracts, and created a taxonomy of test setups for simulation-based contract compliance checking, the system is finally ready for contract-based system verification. The goal of this step is to prove that all refinement steps from the top-level contract down to the atomic component contracts are correct, that the contracts of all connected components are compatible, and that all components comply with their contract. If this is achieved, we have produced substantial verification evidence to show that the implementation of the system meets the top-level contract. If the top-level contract is sufficiently complete, this ensures that the autonomous vessel exhibits safe and correct interactions with its operative environment. If the verification fails at some stage, a design change is required.

We split the contract-based verification procedure into three main activities, which will be applied recursively down through the component structure:

- Contract compliance checking
- Contract composition
- Refinement checking

We first show the derivation of how to perform these activities using the contract framework of Section 2. Then, we present the recursive procedure to be applied in Step 4, which combines these activities.

4.1 Derivation of contract-based verification activities

Contract compliance checking has already been briefly introduced in Section 3. This involves verifying that a component complies with its contract by simulation-based testing with the test setup defined in Step 3. Stated more formally, contract compliance checking involves verifying that all behaviors σ of a component with contract $C = (\varphi_A, \varphi_G)$ satisfies $\varphi_A(\sigma) \rightarrow \varphi_G(\sigma)$.

Contract composition involves composing the contracts for all sub-components into one composite contract. Suppose the component M with contract $C = (\varphi_A, \varphi_G)$ is implemented by k sub-components M_1, M_2, \dots, M_k with contracts C_1, C_2, \dots, C_k , such that $M = M_1 \times M_2 \times \dots \times M_k$. Recall that the composition of two saturated assume-guarantee contracts $C_i = (A_i, G_i)$ and $C_j = (A_j, G_j)$ is defined in terms of sets of behaviors as $C_i \otimes C_j = (A_{C_i \otimes C_j}, G_{C_i \otimes C_j})$, with

$$\begin{aligned} A_{C_i \otimes C_j} &= (A_i \cap A_j) \cup (G_i \cap G_j)^c \\ G_{C_i \otimes C_j} &= G_i \cap G_j. \end{aligned}$$

Transforming this from set-notation to first-order logic formulas using the translations given in Section 2.2.3 yields the composite contract

$$\begin{aligned} \varphi_A^{C_i \otimes C_j} &= (\varphi_A^i \wedge \varphi_A^j) \vee \neg(\varphi_G^i \wedge \varphi_G^j) \\ \varphi_G^{C_i \otimes C_j} &= \varphi_G^i \wedge \varphi_G^j. \end{aligned}$$

Also, recall that contract composition is associative and commutative, that is, $C_1 \otimes (C_2 \otimes C_3) = (C_1 \otimes C_2) \otimes C_3$ and $C_1 \otimes C_2 = C_2 \otimes C_1$. This means that we can construct the logic formula for the composite of any number of contracts using the above formulas by starting with C_1 and incrementally composing it with new contracts. For k contracts, the composite contract $C_{comp} = (\varphi_A^{comp}, \varphi_G^{comp})$ is thus constructed as $C_{comp} = (\dots(((C_1 \otimes C_2) \otimes C_3) \otimes C_4) \dots) \otimes C_k$.

Refinement checking involves checking that the composite contract, C_{comp} , for a set of sub-components refine the contract of the component which they implement. In mathematical terms, this involves checking that

$$C_{comp} = C_1 \otimes C_2 \otimes \dots \otimes C_k \leq C.$$

We propose to formulate refinement checking as a theorem proving problem and feed it to the Z3 ATP. Recall that $C_2 \leq C_1$, iff $A_1 \subseteq A_2$ and $G_2 \subseteq G_1$. Translating this to first-order logic formulas and substituting the composite contract C_{comp} for C_1 and the contract of the parent component C for C_2 yields the following two theorems which must be proved for refinement checking

$$\begin{aligned} \varphi_A &\rightarrow \varphi_A^{comp} \\ \varphi_G^{comp} &\rightarrow \varphi_G \end{aligned}$$

Although it may not be obvious, this method of refinement checking also checks that the contracts of sub-components are compatible with each other. We will briefly demonstrate why the refinement check will fail if there are incompatible contracts among the sub-components. When the pair contracts (C_i, C_j) are not compatible, then the assumptions of the composite contract, φ_A^{comp} will have assumptions from φ_A^j which are not covered by φ_A^i . Moreover, these uncovered assumptions can not be covered by the assumptions of the parent component, M , since they are part of an internal port connection between sub-components and therefore not part of the interface of M . The refinement proof of $\varphi_A \rightarrow \varphi_A^{comp}$ will therefore fail, since there will be cases where φ_A holds but φ_A^{comp} does not.

4.2 The recursive contract-based verification procedure of Step 4

Next, we state the recursive procedure used in Step 4 which combines these three activities to achieve system verification. The procedure is shown in Algorithm 1.

We formulate the contract-based verification process as a recursive procedure, *verifyComponent()*. Step 4 simply involves applying this procedure to the top-level component, and it will recursively call itself for all sub-components. The recursion breaks when the procedure is applied to atomic components. The input to the procedure is a component M with contract $C = (\varphi_A, \varphi_G)$. The first step of the procedure is the contract compliance checking, which involves verifying that all behaviors σ of M satisfy $\varphi_A(\sigma) \rightarrow \varphi_G(\sigma)$ using methodology of choice. After this, the procedure branches depending on whether the component is composite or atomic. If it is atomic, the recursion breaks and the procedure terminates. If it is composite, the procedure proceeds to verify the sub-components which implement M . First, the contracts of all sub-components are saturated. Then, the composite contract $C_{comp} = (\varphi_A^{comp}, \varphi_G^{comp})$ of all sub-component contracts is incrementally constructed. Refinement checking is achieved by applying Z3's *prove* function to the theorems $\varphi_A \rightarrow \varphi_A^{comp}$ and $\varphi_G^{comp} \rightarrow \varphi_G$. Finally, the procedure recursively calls itself with each sub-component of M as the arguments. This will do the contract compliance check for all sub-components against their respective contracts, and continue to verify their respective sub-component structures if they are composite. Hence, by applying Algorithm 1 to the top-level component, the entire component tree of the system will be verified.

Algorithm 1: *verifyComponent(Component M)*

```

input: Component  $M$  with contract  $C = (\varphi_A, \varphi_G)$ 
// Contract compliance checking
Verify that all behaviors  $\sigma$  of  $M$  satisfies  $\varphi_A(\sigma) \rightarrow \varphi_G(\sigma)$  using methodology of choice;
// If the  $M$  is composite, verify the sub-component structure
if  $M$  is composite then
  // Saturate all sub-component contracts  $C_1, C_2, \dots, C_k$ 
  for  $i = 1:k$  do
    |  $\varphi_G^i = \varphi_G^i \vee \neg\varphi_A^i$ ;
  end
  // Composition of the sub-component contracts  $C_1, C_2, \dots, C_k$ 
   $C_{comp} = (\varphi_A^{comp}, \varphi_G^{comp}) = C_1$ ;
  for  $i = 2:k$  do
    |  $\varphi_A^{comp} = (\varphi_A^{comp} \wedge \varphi_A^i) \vee \neg(\varphi_G^{comp} \wedge \varphi_G^i)$ ;
    |  $\varphi_G^{comp} = \varphi_G^{comp} \wedge \varphi_G^i$ ;
  end
  // Refinement checking
   $\text{prove}(\varphi_A \rightarrow \varphi_A^{comp})$ ;
   $\text{prove}(\varphi_G^{comp} \rightarrow \varphi_G)$ ;
  // Recursively apply verifyComponent() to each sub-component of  $M$ 
  for each sub-component  $M_i$  of  $M$  do
    |  $\text{verifyComponent}(M_i)$ ;
  end
end
return

```

4. Case study: The milliAmpere II autonomous passenger ferry

In this section, we demonstrate the use of our contract-based verification framework in a case study with the autonomous passenger ferry milliAmpere II.

4.1. Description of the vessel and operation

milliAmpere II is a small autonomous double-ended ferry designed for carrying 12 pedestrians and cyclists across the canal in Trondheim, Norway. The milliAmpere II is a follow-up on the research prototype milliAmpere (Brekke, Eide, Eriksen, Wilthil, Breivik, Skjellaug, Helgesen, Lekkas, Martinsen, Thyri, Torben, Veitch, Alsos and Johansen, 2022). In contrast to its predecessor, milliAmpere II will be put into regular passenger traffic and is therefore designed in accordance with the national regulations for passenger transport (NMD, 1990). The ferry is owned and will be operated by the Norwegian University of Science and Technology (NTNU).



Figure 6: The milliAmpere II ferry docking at the Ravnkloa side of the Trondheim canal. Photo: Egil Eide.

milliAmpere II has length 8.5m and beam 3.5m. It has a fully electric propulsion system with induction charging and four 10kW thrusters with controllable azimuth angles. The service speed is 5 knots. The ferry is equipped with a class-approved motion control and automation system delivered by Marine Technologies. The navigation system of the ferry is a GNSS-aided inertial navigation system (INS) from SentiSystems with real-time kinematic (RTK) corrections. This system provides 6DOF motion measurements with centimeter-level positioning. The navigation system also features dead reckoning capabilities to ensure safe return to quay in case of a GNSS outage. The NTNU spin-off company Zeabuz has developed the high-level autonomy system. The ferry utilizes a range of different exteroceptive sensors for obtaining situational awareness. It is equipped with a top-mounted Simrad maritime radar, FLIR Boson infrared cameras, and FLIR Blackfly S RGB cameras, as well as Ouster OS1 lidars placed at the corners of the ferry. The autonomy system is responsible for processing the sensor data and for providing the motion control system with a collision-free motion reference. The ferry will travel along a fixed pre-planned path and only control the speed along that path to avoid collisions with other vessels. The collision avoidance algorithm is described in Thyri, Breivik and Lekkas (2020)

The area of operation is the stretch from Ravnkloa to Vestre Kanalkai. The crossing is approximately 85m long, which is expected to take about one minute at normal service speed. The area is regulated as fully enclosed waters, with a speed limit of 5 knots (2.6 m/s). The maximum wave height in the area is 0.5m and the maximum current speed is 3 knots (1.5 m/s). The area may be subject to harsh weather, and the ferry will be suspended if the wind speed exceeds

10 m/s. There are no shallows and no static obstacles in the area, apart from the sides of the canal and boats that are moored to the side of the canal. The traffic in the canal consists of motorized vessels of varying sizes and types. In addition, the canal is subject to heavy kayak and canoe traffic during the summer season. A bird's eye view of the area is given in Figure 7.

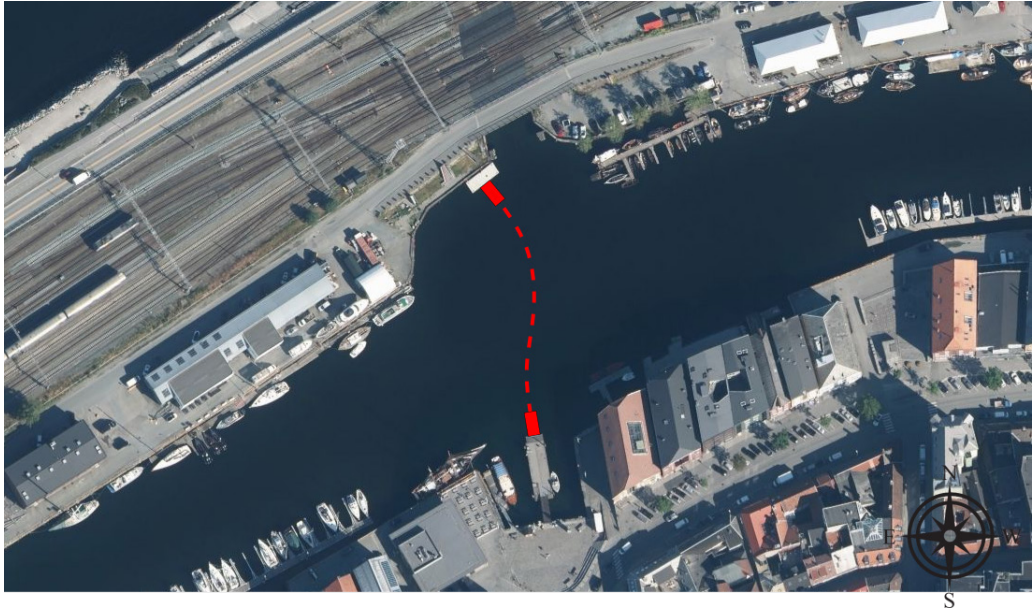


Figure 7: The area of operation for milliAmpere II. The crossing is an 85m long stretch between Ravnkloa and Vestre Kanalkai across the canal in Trondheim, Norway.

4.2. Applying the contract-based verification methodology

Having introduced the object of the case study, we proceed to follow the steps in the contract-based verification methodology introduced in Section 3. The full component structure we develop in the case study is given in Figure 8.

Step 1: Define the top-level component and contract

We start at the top-level view and define the interaction with the operative environment. As described in Section 3, this should be done by a systematic hazard identification process. To demonstrate the main features of the methodology in a brief and concise manner, we limit the scope to studying only a couple of key interactions with the operative environment. We consider the interactions with a moving obstacle and the environmental loads. We define an *obstacle_motion* out-port on the operative environment which carries messages of the *VesselMotionMsg* type:

```
class VesselMotionMsg:
    def __init__(self, name):
        self.speed = Real(name + '_speed')
        self.course = Real(name + '_course')
        self.position = RealVector(name + '_position', 2)
```

The value on the *obstacle_motion* port will be *None* if there are no visible obstacles. We also define an in-port on the operative environment for the milliAmpere II motion, as the motion of milliAmpere II may influence the behavior of the obstacle. We name this port *ferry_motion* and it also carries messages of the *VesselMotionMsg* type. Finally, we define an out-port named *environment_loads* with message type *EnvLoadMsg* defined as:

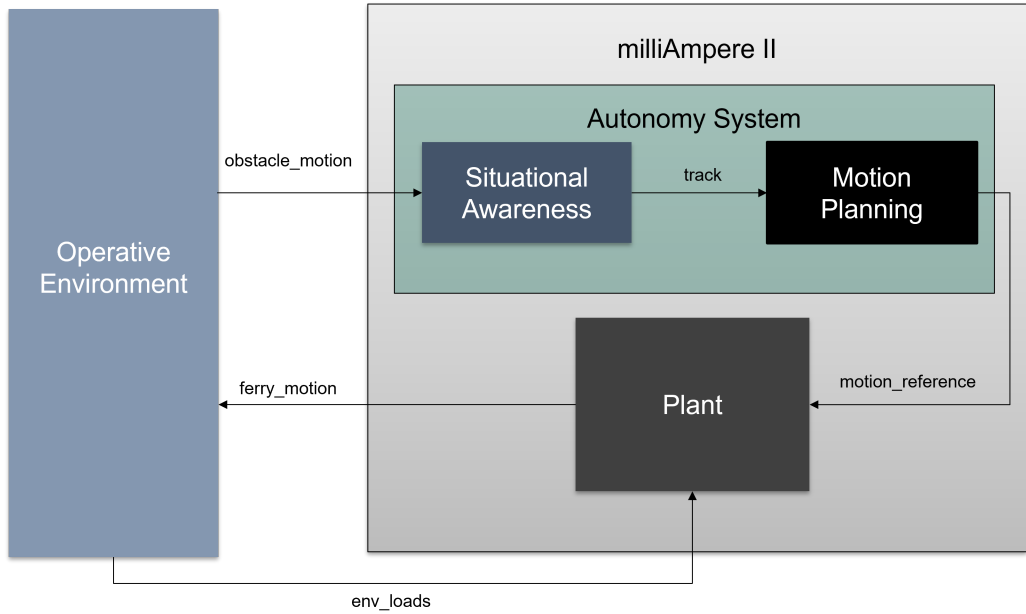


Figure 8: The component structure used in the case study.

```
class EnvLoadMsg:
    def __init__(self):
        self.current_speed = Real('current_speed')
        self.wind_speed = Real('wind_speed')
        self.wind_direction = Real('wind_direction')
        self.sig_waveheight = Real('sig_waveheight')
        self.peak_period = Real('peak_period')
```

The current direction is assumed to be parallel to the canal, and the direction of flow is given by the sign of the current speed, where a positive speed represents downstream current (from west to east in Figure 7).

Having defined the environmental interactions, the next step is to define the top-level contract. As Figure 4 showed, this contract plays a fundamental role, as its assumptions form the ODD and the guarantees form the top-level requirements to ensure safe interactions with the operative environment. We define the following top-level contract:

```
mA2_assumptions = And(
    obstacle_motion.speed >= 0,
    obstacle_motion.speed <= 2.6,
    env_loads.current_speed >= -1.5,
    env_loads.current_speed <= 1.5,
    env_loads.wind_speed >= 0,
    env_loads.wind_speed <= 10.0,
    env_loads.wave_height >= 0,
    env_loads.wave_height <= 0.5)

mA2_guarantees = And(
    distance_2d(obstacle_motion.position, ferry_motion.position) >= 10.0,
    distance_2d(ferry_motion.position, path) <= 1.0)
```

The assumptions are derived from the CONOPS for milliAmpere II. The assumptions specify that the obstacles keep the speed limit for fully enclosed waters of 5 knots (2.6m/s), the current speed is less than 1.5m/s, the wind speed is less than 10m/s, and that the significant wave height is less than 0.5m. For the guarantees of the top-level contract, we limit the focus to avoiding collisions with the moving obstacle and following the pre-planned path. The guarantees specify

that the ferry's distance to the moving obstacle should be greater than 10m and that the ferry does not deviate from the pre-planned path by more than 1m. Since there are no static obstacles within these margins, these two guarantees ensure that there are no collisions. Other aspects could have been included here, such as passenger comfort by constraining the allowed speed and acceleration. Such aspects are excluded for the sake of brevity but would be incorporated in the contract in the same way. The *Z3 And* operator builds a logic formula which is the conjunction of all the arguments. The *path* variable is a 2-dimensional vector whose value is equal to the point on the pre-defined path which is closest to *ferry_motion.position* at all times. The *distance_2d* function is defined as:

```
def distance_2d(pos1, pos2):
    return Sqrt((pos1.north-pos2.north)**2 + (pos1.east-pos2.east)**2)
```

Before we conclude Step 1, we refer back to Section 2 to show how these concrete component and contract definitions relate to the mathematical notation. At the top-level view, the variables v_1, v_2, \dots, v_{11} correspond to the 11 variables in the message type definitions of the *obstacle_motion*, *ferry_motion* and *env_loads* ports. That is, $v_1 = \text{obstacle_motion.speed}$, $v_2 = \text{obstacle_motion.course}$, ..., $v_{11} = \text{env_loads.peak_period}$. The numbering is chosen arbitrarily in this example. The domains of each variable, D_1, D_2, \dots, D_{11} correspond to the data types of these variables, as defined in the message types. In this case, $D_1 = D_2 = \dots = D_{11} = \mathbb{R}$, and $V = D_1 \times D_2 \times \dots \times D_{11} = \mathbb{R}^{11}$. At each time step, the implementation of the components assigns values to each of these variables, resulting in a reaction $s \in \mathbb{R}^{11}$. The discrete-time sequence of reactions is the behavior, σ , at this abstraction level. Let $C_{mA2} = (\varphi_A, \varphi_G)$ denote the top-level contract. The *mA2_assumptions* specify the first-order logic formula φ_A and *mA2_guarantees* specifies φ_G . Each of these formulas is built by combining predicates over the variables v_1, v_2, \dots, v_{11} with first-order logic operators, in this case, the *And* operator.

Step 2: Stepwise refinement into sub-components

Having defined the top-level component and contract, in Step 2 we proceed by refining them into a set of sub-components that implement the top-level component. As Figure 8 shows, in the first refinement steps we have chosen to split the top-level component (milliAmpere II) into two sub-components; the autonomy system and the plant, in a standard controller-plant configuration. The plant component contains the physical ferry and the low-level control functionality, such as the industrial motion control system, the actuator control, and the navigation system.

The interaction between the autonomy system and the plant takes place over the *motion_reference* port, where the autonomy system provides the plant with the trajectory that the motion control system should track. We define the *motion_reference* port to also send messages of the *VesselMotionMsg* type.

Next, we need to define the contracts at this abstraction level. The contracts need to refine the contract for the top-level component and be compatible across the two components. Additionally, the contracts may also need to consider any inherent limitations in the sub-components, such as the precision of the motion control system. We begin by defining the contract for the plant. The assumptions on the environmental loads are simply copied from the top-level component. The assumptions on the motion reference constrain the speed reference to be between -2m/s and 5m/s, to receive a feasible trajectory to track. The guarantee of the plant specifies that the trajectory error of the motion control system should be less than 1m.

```
plant_assumptions = And(env_loads.current_speed >= -1.5,
                        env_loads.current_speed <= 1.5,
                        env_loads.wind_speed >= 0,
                        env_loads.wind_speed <= 10.0,
                        env_loads.wave_height >= 0,
                        env_loads.wave_height <= 0.5,
                        motion_ref.speed >= -2.0,
                        motion_ref.speed <= 5.0)

plant_guarantees = distance_2d(motion_ref.position, ferry_motion.position) <= 1.0
```

Next, we define the contracts for the autonomy system. The assumptions on the traffic port are copied directly from the top-level component. The guarantees specify that the motion reference needs to maintain a safe distance of 11m from the obstacle. The safe distance from the ferry to the obstacle in the top-level contract was only 10m, however, since

the plant component can only track the motion reference from the autonomy system with a precision of 1m, the safe distance guarantee for the motion reference needs to be increased to 11m for the autonomy system and plant contracts to correctly refine the top-level contract. Additionally, we specify that the motion reference should coincide with the pre-planned path at all times, meaning that we only allow the autonomy system to generate references along this path, effectively meaning it is only controlling the ferry speed and position along the path. To be compatible with the plant contract, the autonomy system also needs to constrain the speed of the motion reference. The contract specifies that the speed reference will be between -1m/s and 3m/s, in order to keep well within the limitations in the plant assumptions.

```
autonomy_assumptions = And(obstacle_motion.speed >= 0.0,
                           obstacle_motion.speed <= 2.6)

autonomy_guarantees = And(distance_2d(motion_ref.position, path) == 0,
                           distance_2d(motion_ref.position, obstacle_motion.position) >= 11,
                           motion_ref.speed >= -1.0,
                           motion_ref.speed <= 3.0)
```

Finally, we will do one more refinement step on the autonomy system component. As Figure 8 shows, we implement the autonomy system component by two sub-components; motion planning and situational awareness. The situational awareness component senses other vessels with its exteroceptive sensors and estimates their position, speed, and heading. This information is sent to the motion planning component over the *track* port. The motion planning component generates a *motion_reference* signal such that the ferry keeps a safe distance from the obstacle track.

The assumptions on the traffic port of the situational awareness component are copied from the autonomy system component. The guarantee of the situational awareness component specifies that if there is a visible obstacle, it will be detected and tracked with an error of less than 3m:

```
sitaw_assumptions = And(obstacle_motion.speed >= 0,
                        obstacle_motion.speed <= 2.6)

sitaw_guarantees = Implies(obstacle_motion is not None,
                           distance_2d(track.position, obstacle_motion.position) <= 3.0)
```

For the *motion_planning* component, we define no assumptions on the *track* port. The guarantees specify that the motion reference should maintain a safe distance to the obstacle track received by the situational awareness system. However, due to the added uncertainty of 3m in the obstacle track introduced by the situational awareness system, the safety margin of the motion reference system needs to be increased from 11m to 14m to refine the autonomy system contract. The guarantees for the motion reference to coincide with the pre-planned path and the constraints on the speed reference are copied from the autonomy system component in order to refine its contract.

```
moplan_assumptions = True

moplan_guarantees = And(distance_2d(motion_ref.position, track.position) >= 14.0,
                        distance_2d(motion_ref.position, path) == 0,
                        motion_ref.speed >= -1.0,
                        motion_ref.speed <= 3.0)
```

Step 3: Define test setups for simulation-based testing

Next, we proceed to Step 3 of the methodology, where the test setups for simulation-based contract compliance checking are defined. For the milliAmpere II case, there already exists an extensive set of simulators used for development and verification. The simulators are developed in a collaboration between NTNU, the autonomous ferry start-up Zeabuz and the classification society DNV in the TRUSST research project (RCN, 2021). We will use these simulators as a basis for this section, and show how they can be configured to define test setups for contract compliance checking.

An overview of the full simulator setup for milliAmpere II is given in Figure 9. The simulator is implemented over several different platforms. The traditional maritime simulation models are implemented on the Open Simulation Platform (OSP) (Smogeli et al., 2020), which is an open standard for co-simulation of maritime systems built on the FMI standard (Blockwitz, Otter, Akesson, Arnold, Clauss, Elmqvist, Friedrich, Junghanns, Mauss, Neumerkel, Olsson

and Viel, 2012). The 3D rendering of the virtual world and sensor models for the exteroceptive sensors are simulated on Gemini, which is an open-source simulation platform for autonomous vessels (Vasstein, Brekke, Mester and Eide, 2020). Gemini is built on the Unity game engine, and its exteroceptive sensor models are based on a novel methodology for simulating electromagnetic radiation sensors using game engine technology (Vasstein et al., 2020). The autonomy software runs on the ROS, and the simulator supports SiL simulation using exact replicas of the autonomy software. This is also true for the COTS motion control system, however, this runs on the proprietary platform of the vendor. Some snapshots from the simulator in action are given in Figure 10.

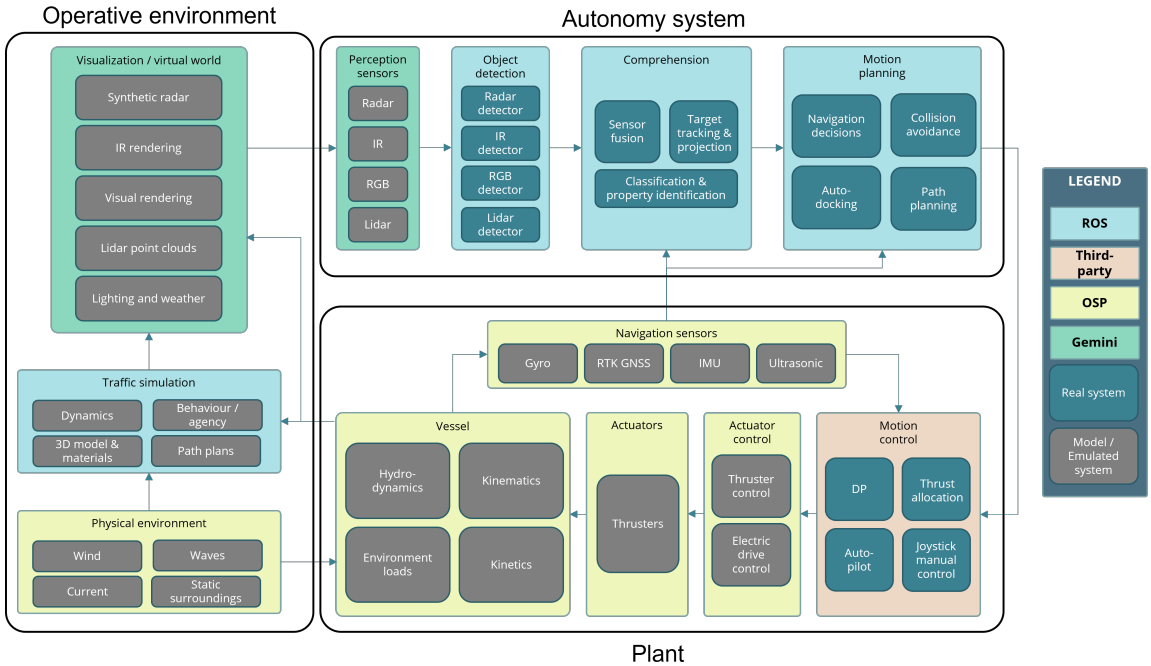


Figure 9: Overview of the milliAmpere II simulator. The black boxes indicate which parts of the simulator correspond to the component structure we have defined in Figure 8. In the autonomy system, the perception sensors, object detection, and comprehension modules together make up the situational awareness component.

Recall that the goal of the simulation-based testing of a given component is to verify that the component complies with its contract, as illustrated in Figure 5. To achieve this, we need a simulation model of the component under test and a simulated test environment to generate the test signals on the in-ports of the component. Together, this defines the test setup for the component. We believe that it is advantageous to keep the test environment of the component as simple as possible in order to focus the testing to find inherent weaknesses in the component under test. This both simplifies the simulator and reduces the size of the test space for the component. Testing components in more realistic environments is of course also very important, however, this will be done when testing at the higher abstraction levels, where the component will be integrated with other components and thus be tested in a more realistic environment.

The full setup of Figure 9 could represent the test setup for the top-level component since it simulates all parts of the component and its operative environment. To define the test setup for components at lower abstraction levels, we can use this setup as a basis, remove all modules which are deemed irrelevant and simplify all connected modules as much as possible. Figure 11 shows the test setup for contract compliance checking of the autonomy system component. The vessel motion of the ferry is set equal to the motion reference output of the motion planning component, thereby we are assuming perfect trajectory tracking of the plant. The main objective of this setup is to test the integrated

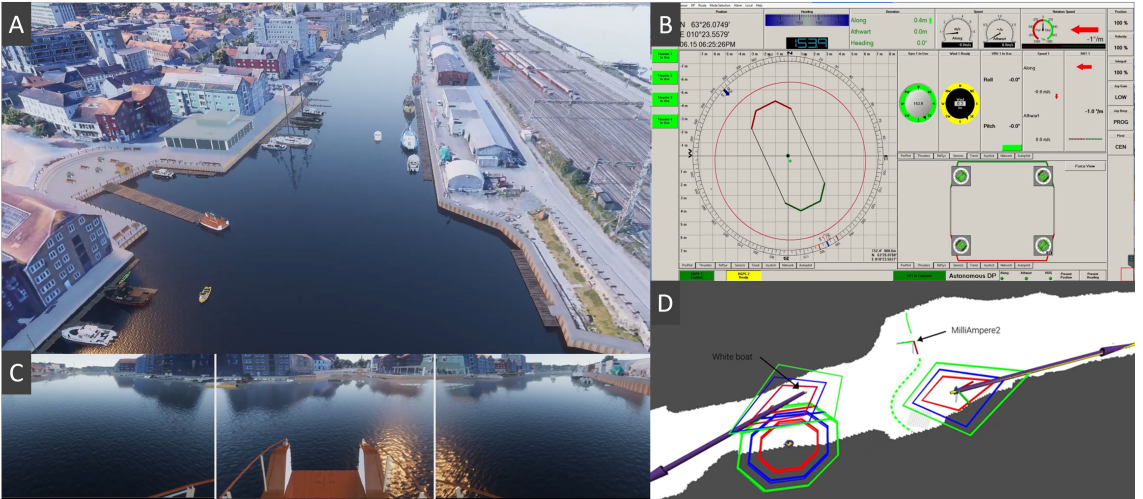


Figure 10: Snapshots from the milliAmpere II simulator. A) The 3D rendering of the operation area and the ferry. B) The interface for the motion control system, including DP, thruster control, and navigation. C) The output of the simulated RGB cameras. D) A map of the situational awareness, where the detections from the simulated camera, lidar and radar sensors are fused and used to estimate the pose and speed of obstacles. Also shown in D are the safety margin zones around each obstacle.

motion planning and situational awareness system for collision avoidance. Figure 12 shows the test setup for contract compliance checking of the plant component. The main objective of this setup is to test the trajectory tracking capabilities of the plant under different environmental loads and reference trajectories. To this end, we have defined a *test trajectories* module which generates motion reference inputs to the plant. Figure 13 shows the test setup for contract compliance checking of the motion planning component. In this setup we are simplifying the test environment to use perfect tracks and perfect trajectory tracking, thus reducing both the situational awareness system and the plant to identity/pass through. The main objective of this setup is to verify collision avoidance for the motion planning component. Being very lightweight, this setup offers fast simulations and corresponding high test coverage. Finally, the test setup for the situational awareness component is shown in Figure 14. The main objective of this setup is to verify the obstacle tracking capabilities of the situational awareness component. We are using the *test trajectories* module to generate the ferry motion.

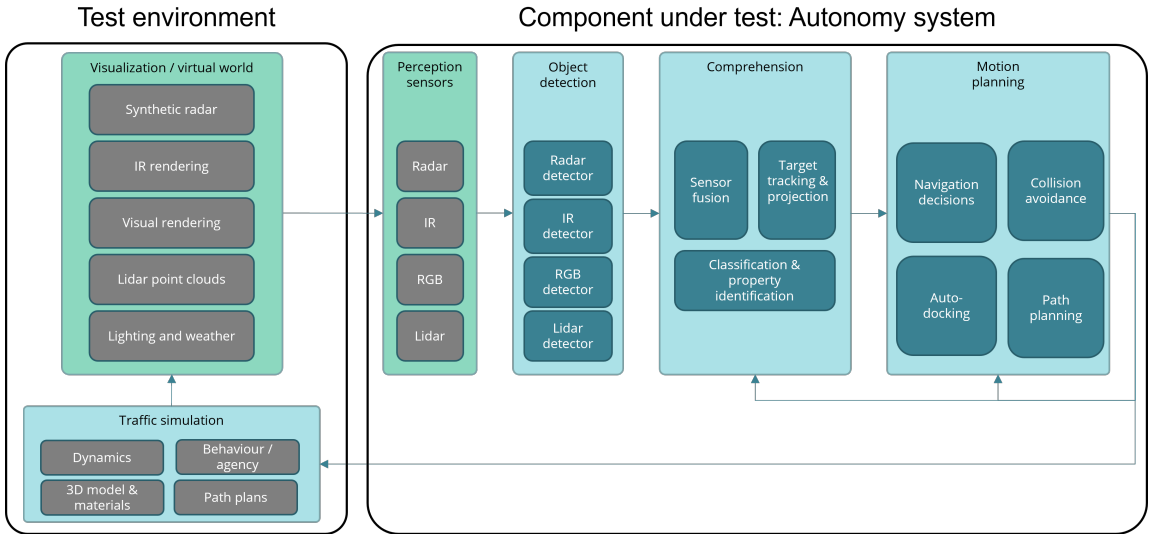


Figure 11: Test setup for contract compliance checking of the autonomy system component.

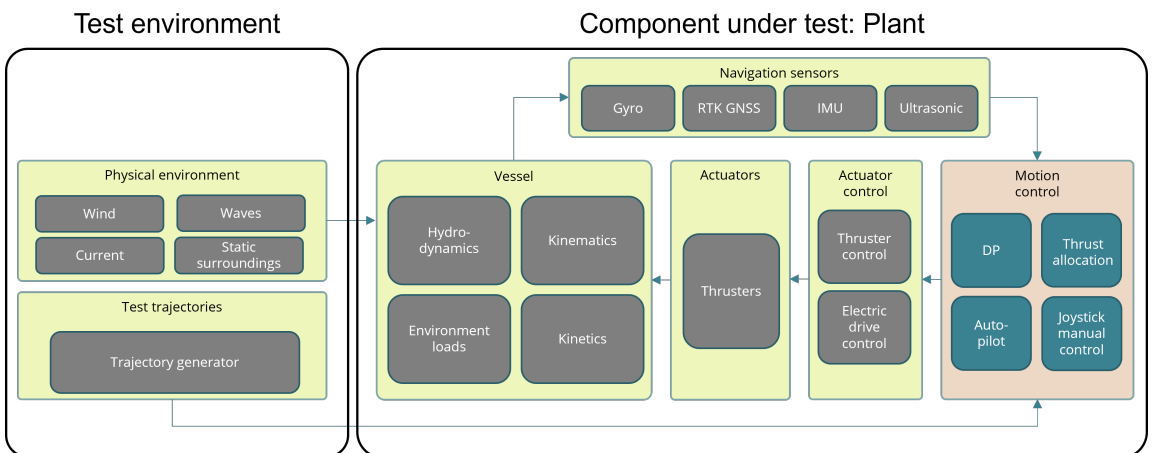


Figure 12: Test setup for contract compliance checking of the plant component.

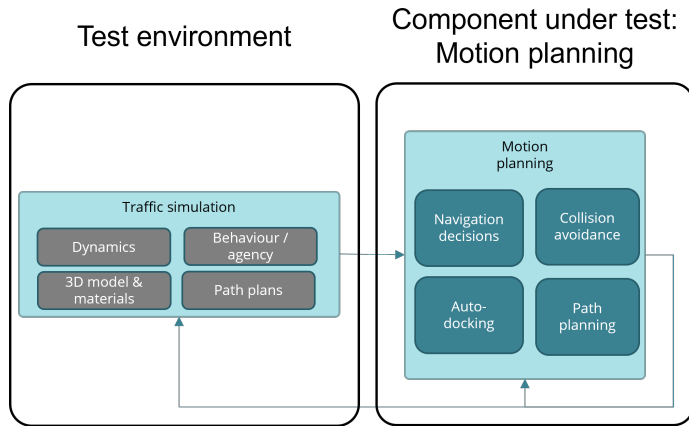


Figure 13: Test setup for contract compliance checking of the motion planning component.

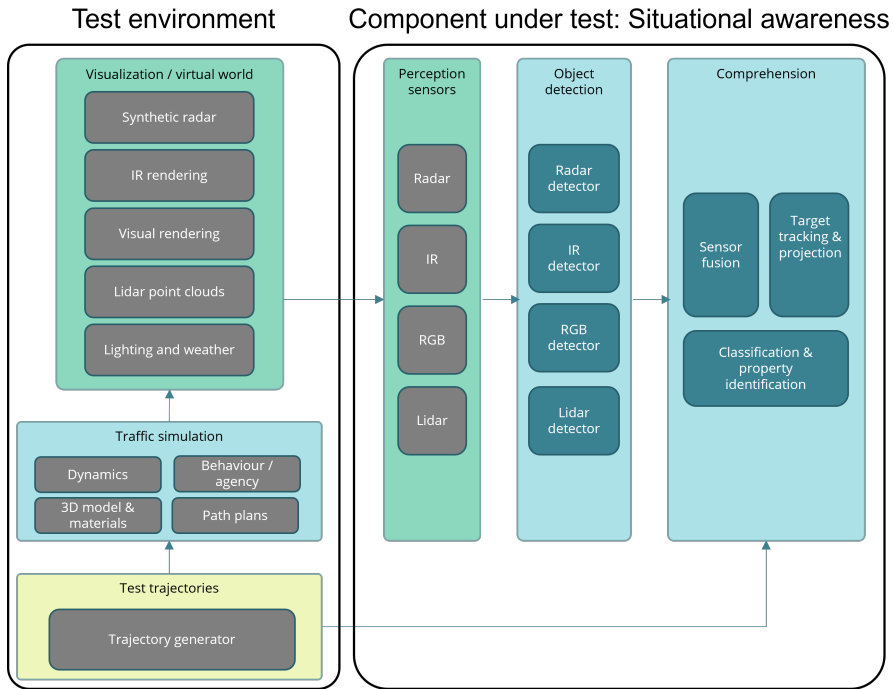


Figure 14: Test setup for contract compliance checking of the situational awareness component.

Step 4: Recursive contract-based verification

With the component structure, contracts, and test setup in place, we can readily apply the recursive contract-based verification procedure of Algorithm 1 to the milliAmpere II system to achieve contract-based system verification. To demonstrate contract-based verification, we will in this section go through the step-by-step computations of Algorithm 1 applied to the components and contracts developed in the case study.

The entry point for the verification is applying *verifyComponent()* on the top-level milliAmpere II component. The first step is contract compliance checking. This involves running simulations with the test setup of Figure 9 to verify that all behaviors σ satisfy $\varphi_A(\sigma) \rightarrow \varphi_G(\sigma)$. This amounts to running scenarios where the obstacle motion and environmental loads are within the assumptions and verify that the ferry keeps a 10m distance to the obstacle and tracks the path with 1m precision. After the contract compliance check is completed, the procedure continues into the branching statement of Algorithm 1, since the top-level component is composite. First, the composite contract of the sub-components is constructed by composing the contracts of the autonomy system and the plant. Then, the procedure performs the refinement check by passing the theorems $\varphi_A \rightarrow \varphi_A^{comp}$ and $\varphi_G^{comp} \rightarrow \varphi_G$ to Z3's *prove()* function. This yields a *proved* result and thereby verifies that the composite contract of the autonomy system and plant refine the top-level contract and that the contracts of the autonomy system and plant are compatible.

The next step in Algorithm 1 is to recursively apply *verifyComponent()* to the sub-components of the top-level component. The procedure begins by applying it to the plant component. First, the contract compliance check is performed, this time using the test setup of Figure 12. After completing the contract compliance checking the recursion breaks, since the plant is an atomic component. The procedure then moves on to applying *verifyComponent()* to the autonomy system component. This begins with contract compliance checking using the test setup of Figure 11. Since the autonomy system component is composite, the procedure continues into the branching statement of Algorithm 1, which composes the contracts for the situational awareness and motion planning components and checks that the composite contract refines the autonomy system contract. Passing refinement checking theorems to Z3's *prove()* function again yields a *proved* result.

Finally, we recursively apply *verifyComponent()* to the situational awareness and motion planning components. The contract compliance checks are executed using the test setups of Figures 14 and 13, respectively. Since both these components are atomic, the recursion breaks and Algorithm 1 terminates. The entire formal reasoning for the case study, including definition, saturation, and composition of contracts in addition to the refinement proofs, was executed in 4.9 seconds on a laptop with an Intel Core i9-9980HK CPU.

5. Discussion

The results from the case study indicate that the proposed contract framework and verification methodology overall achieve the goal of modular verification. Next, we discuss some challenges we encountered and some possible solutions, as well as directions for future work.

We experienced the Z3 first-order logic language as a natural and expressive syntax for specifying contracts. However, we encountered some difficulty in the automatic theorem proving of the refinement checks. When calling Z3's *prove()* function, a successful run returns either *proved* or *could not prove* with a counterexample. However, in some cases, it returns *unknown* or runs seemingly indefinitely without returning a result. In these cases, it can be hard and time-consuming to debug. It is well known that Z3 can be sensitive to seemingly unimportant changes in the theorem formulation, such as the order in which arguments are given. Some trial and error can therefore be necessary to get a successful run. For proofs including boolean variables and linear constraints over real variables, Z3 appeared to be very capable and scalable. The proofs involving nonlinear functions over real-valued variables did, however, also pose some difficulty. An example of this is the proofs involving the *distance_2d()* function, which is nonlinear due to the quadratic terms and the square root operation. It was not unexpected that this was problematic, as automatic theorem proving of such problems is a very hard computational problem that is undecidable in general. To get successful proofs here, we sometimes had to manually split the proofs into smaller and more manageable pieces which were proved independently. Z3 offers functionality for the user to input proof tactics, which may increase the capabilities to prove hard problems. An alternative approach to enhance the capabilities to prove nonlinear real-valued theorems is to use

an interactive theorem prover, such as Isabelle (Paulson, 1994) or Coq (Bertot, 2008). These are generally capable of proving more complex problems, but also require more expertise and input from the user.

In this work, a critical success factor for a contract-based verification methodology is that it is scalable enough to give value to complex industrial autonomy use cases. In order to present a case study in full in this paper, the case study had to be simplified in several aspects. The components were simplified versions of the real components in the milliAmpere II system and not all interactions between components were considered. The operative environment model was also simplified. The contracts only considered certain aspects of the verification scope, and important aspects such as passenger comfort, reliability, fault handling, and timing were not considered. The refinement of the components was also stopped at a quite high level of abstraction. Because of these simplifications, the scalability of the methodology to industrial use cases can not be concluded based on the case study only. Our intuition is, however, that such a structured approach is necessary to manage the complexity of autonomous vessel systems, and that scalability is, to a large degree, dependent on having sufficient tool support for defining and managing a large number of components and contracts in an enterprise setting. For the time being, the biggest obstacle to scalability seems to be the formal compositional reasoning, as discussed above. We do, however, believe that having a structured framework for design and verification using the concepts of assume-guarantee contracts, refinement, and composition, can provide great value to the design and verification process, even if the contracts are expressed in natural language and the compositional reasoning is informal.

Another motivation for doing modular verification is that it enables more efficient change management. Since autonomous vessels are highly software intensive they will likely be subject to frequent software updates. After a software update, the system will need to be re-verified. Re-verifying the entire system is a huge task, and doing this for every update is not desirable. The recursive contract-based procedure of Algorithm 1 can be extended to support selective re-verification, that is, only re-verify the parts of the system which are affected by the change. A straightforward approach to this is to add a hash to each component. The hash should be computed based on the component implementation, port definition, parameters, and contract. For composite components, the hash should also be computed based on the hash of all sub-components. The *verifyComponent(Component M)* procedure can then check if the hash of the *M* has changed since the last verification and terminate immediately if it has not changed. Suppose for instance a software update has been made to the situational awareness component in Figure 8. This would change the hash of the situational awareness, autonomy system, and milliAmpere II components, but not the motion planning and plant components. The re-verification would thus re-verify the situational awareness component, its integration into the autonomy system, and its integration into the milliAmpere II component, but would skip the re-verification of the motion planning and plant components. If the contract compliance checking for the changed component is exhaustive, it would be sufficient to only re-verify that component. However, this will in general not be the case when using simulation-based contract compliance checking.

As introduced in Section 1, a key challenge for verifying autonomous systems is that they extensively sense and interact with the open environment, and there will be an infinite number of scenarios that are unknown during design. Such scenarios are often referred to as *edge cases*. This challenge is well-known from automotive autonomy, which has approached the challenge by extensive use of machine learning, with the hope that vast amounts of training data will expose the self-driving system to a sufficient number of edge cases. This approach has not yet been successful (Koopman, 2021). We believe that contracts may offer an alternative solution. While it is very hard to design an autonomous system that will act correctly in all edge cases, it may be feasible to design a system that can detect when a situation is outside the ODD and do something safe, such as going to a minimum risk condition (MRC). Compared to the automotive application, the safety margins generally are larger, and the speed is generally lower in maritime applications. Therefore, it may be possible to ensure safety by going to an MRC. Online monitoring of the assumptions of the top-level contract, which specify the ODD, can implement such a system. We therefore consider online monitoring of assumptions and the use of this in the high-level decision making as an important topic for future work.

Finally, there are some directions in which the methodology can be extended. As mentioned in Section 2.1, we only consider synchronous behaviors. The extension to asynchronous behavior is a natural next step. Extensions to more expressive contracts is also a possible direction for future work. Examples include temporal contracts, using e.g. a temporal logic, and the use of quantifiers in the first-order logic formulas. Using the concept of different *viewpoints*,

as proposed by Nuzzo et al. (2015), to combine different orthogonal aspects in the contracts is another interesting extension. A critical aspect of the proposed methodology is that it is dependent on completeness in the top-level contracts to achieve the verification objective. Furthermore, generating sufficiently many relevant test scenarios for the simulation may be demanding. As we briefly mentioned in Section 3, risk analysis may provide an important basis for defining safety requirements for the system, as well as for developing and selecting the test scenarios. We see this as an important direction for future work.

6. Conclusions and further work

This paper has introduced contract-based methods to the design and verification of autonomous vessels. We have presented a framework for specifying components and contracts based on the automatic theorem prover Z3 and a contract-based verification methodology using this framework. The contract-based verification methodology approaches compositional reasoning by constructing a composite assume-guarantee contract. Using the Z3 automated theorem prover, formal and automatic refinement checking has been achieved. Furthermore, we have shown how simulation-based testing can be used in conjunction with a contract-based framework in a mutually beneficial way. The contract-based verification method was ultimately stated as a concise recursive procedure for system verification. The contract framework and verification procedure were demonstrated in a case study with the autonomous passenger ferry milliAmpere II.

The discussion highlighted the need for further work in the automatic theorem proving of the compositional reasoning. In particular, methods to better handle proofs including nonlinear functions over real variables should be investigated. Use of Z3's proof tactics or interactive theorem provers were suggested as possible directions for future work. The discussion also suggested possible extensions to the methodology, including online monitoring of assumptions and investigating the use of risk analysis methods as a basis for defining the top-level contract and test scenarios. Finally, the discussion highlights the need for demonstrating the use of the contract-based framework in an industrial-scale project to investigate the scalability of the method. Our hope is that our introduction of contract-based methods to maritime autonomy will trigger more research and development in this direction and ultimately contribute to safer and more robust autonomous vessel control systems.

Acknowledgments

The work by T.R. Torben, I.B. Utne, and A.J. Sørensen is partly sponsored by the Research Council of Norway through the Centre of Excellence funding scheme, project number 223254, AMOS, and project ORCAS with project number 280655. The work by Øyvind Smogeli and Jon Arne Glomsrud is sponsored by the Research Council of Norway through the TRUSST project with project number 313921.

References

- Abrial, J.R., 2011. Modeling in event-b: System and software engineering. Cambridge University Press.
- Bakdi, A., Glad, I.K., Vanem, E., 2021. Testbed Scenario Design Exploiting Traffic Big Data for Autonomous Ship Trials Under Multiple Conflicts With Collision/Grounding Risks and Spatio-Temporal Dependencies. *IEEE Transactions on Intelligent Transportation Systems*, 1–17. doi:10.1109/TITS.2021.3095547.
- Benveniste, A., Caillaud, B., Ferrari, A., Mangeruca, L., Passerone, R., Sofronis, C., 2008. Multiple Viewpoint Contract-Based Specification and Design, in: de Boer, F.S., Bonsangue, M.M., Graf, S., de Roever, W.P. (Eds.), *Formal Methods for Components and Objects*, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 200–225.
- Benveniste, A., Caillaud, B., Nickovic, D., Passerone, R., Racllet, J.B., Reinkemeier, P., Sangiovanni-Vincentelli, A., Damm, W., Henzinger, T.A., Larsen, K.G., 2018. *Contracts for System Design*. volume 12. Now.
- Bertot, Y., 2008. A short presentation of Coq, in: Mohamed, O.A., Muñoz, C., Tahar, S. (Eds.), *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 12–16. doi:10.1007/978-3-540-71067-7_{_}3.
- Blockwitz, T., Otter, M., Akesson, J., Arnold, M., Clauss, C., Elmquist, H., Friedrich, M., Junghanns, A., Mauss, J., Neumerkel, D., Olsson, H., Viel, A., 2012. Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models. *Proceedings of the 9th International MODELICA Conference*, September 3-5, 2012, Munich, Germany 76, 173–184. doi:10.3384/ecp12076173.
- Brekke, E.F., Eide, E., Eriksen, B.O.H., Wilthil, E.F., Breivik, M., Skjellaug, E., Helgesen, O.K., Lekkas, A., Martinsen, A.B., Thyri, E.H., Torben, T., Veitch, E., Alsos, O.A., Johansen, A., 2022. milliAmpere: An Autonomous Ferry Prototype. *Journal of Physics: Conference Series* 2311, 012029.

- Chaal, M., Valdez Banda, O.A., Glomsrud, J.A., Basnet, S., Hirdaris, S., Kujala, P., 2020. A framework to model the STPA hierarchical control structure of an autonomous ship. *Safety Science* 132, 104939. URL: <https://doi.org/10.1016/j.ssci.2020.104939>, doi:10.1016/j.ssci.2020.104939.
- Cimatti, A., Tonetta, S., 2012. A property-based proof system for contract-based design. *Proceedings - 38th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2012*, 21–28doi:10.1109/SEAA.2012.68.
- Clarke, E.M., 1997. Model checking, in: Ramesh, S., Sivakumar, G. (Eds.), *Foundations of Software Technology and Theoretical Computer Science*, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 54–56.
- Clarke, E.M., Long, D.E., McMillan, K.L., 1989. Compositional model checking, in: *Proceedings. Fourth Annual Symposium on Logic in Computer Science*, pp. 353–362. doi:10.1109/LICS.1989.39190.
- Foster, S., Gleirscher, M., Calinescu, R., 2020. Towards Deductive Verification of Control Algorithms for Autonomous Marine Vehicles, in: *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems*, pp. 113–118.
- Hake, G., Hohl, C.P., Hahn, A., 2021. Continuous Contract Based Verification of Updates in Maritime Shipboard Equipment URL: <https://doi.org/10.3390/jmse9070688>, doi:10.3390/jmse9070688.
- Kahn, G., 1974. *The Semantics of a Simple Language for Parallel Programming*. Technical Report.
- Kapinski, J., Deshmukh, J.V., Jin, X., Ito, H., Butts, K., 2016. Simulation-Based Approaches for Verification of Embedded Systems. *IEEE Control Systems Magazine* 36.
- Koopman, P., 2021. SOTIF & Edge Cases URL: https://users.ece.cmu.edu/~koopman/lectures/ece642/L103_SOTIF_EdgeCases.pdf.
- Maler, O., Nickovic, D., 2004. Monitoring temporal properties of continuous signals, in: *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, Springer. pp. 152–166.
- Meyer, B., 1992. Applying 'design by contract'. *Computer* 25, 40–51.
- de Moura, L., Bjørner, N., 2008. Z3: An Efficient SMT Solver, in: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 337–340.
- Murray, B., Rødseth, O.J., Nordahl, H., Wennersberg, L.A.L., Pobitzer, A., Foss, H., 2022. Approvable AI for Autonomous Ships: Challenges and Possible Solutions, in: *Proceedings of the 32nd European Safety and Reliability Conference (ESREL 2022)*, pp. 1975–1982. doi:10.3850/978-981-18-5183-4.
- NMD, 1990. Nordisk båtstandard for yrkesbåter under 15 meter. Technical Report.
- Nuzzo, P., Sangiovanni-Vincentelli, A.L., Bresolin, D., Geretti, L., Villa, T., 2015. A Platform-Based Design Methodology With Contracts and Related Tools for the Design of Cyber-Physical Systems. *Proceedings of the IEEE* 103, 2104–2132. doi:10.1109/JPROC.2015.2453253.
- Nuzzo, P., Xu, H., Ozay, N., Finn, J.B., Sangiovanni-Vincentelli, A.L., Murray, R.M., Donzé, A., Seshia, S.A., 2014. A contract-based methodology for aircraft electric power system design. *IEEE Access* 2, 1–25. doi:10.1109/ACCESS.2013.2295764.
- Paulson, L.C., 1994. Isabelle: A generic theorem prover. volume 828. Springer Science & Business Media.
- Pedersen, T.A., Glomsrud, J.A., Ruud, E.L., Simonsen, A., Sandrib, J., Eriksen, B.O.H., 2020. Towards simulation-based verification of autonomous navigation systems. *Safety Science* 129, 104799. doi:10.1016/j.ssci.2020.104799.
- RCN, 2021. TRUSST: Assuring Trustworthy, Safe and Sustainable Transport for All. URL: <https://prosjektbanken.forskningsradet.no/project/FORISS/313921?Kilde=FORISS&distribution=Ar&chart=bar&calcType=funding&Sprak=no&sortBy=date&sortOrder=desc&resultCount=30&offset=210&TemaEmne.2=N\T1\æring+og+handel>.
- Rokseth, B., Haugen, O.I., Utne, I.B., 2019. Safety Verification for Autonomous Ships. *MATEC Web of Conferences* 273, 02002. doi:10.1051/mateconf/201927302002.
- Rokseth, B., Utne, I.B., 2019. Deriving safety requirement hierarchies for families of maritime systems. *Transactions of the Royal Institution of Naval Architects Part A: International Journal of Maritime Engineering* 161, A229–A243. doi:10.3940/rina.ijme.2019.a3.526.
- Sangiovanni-Vincentelli, A., Damm, W., Passerone, R., 2012. Taming Dr. Frankenstein: Contract-based design for cyber-physical systems. *European Journal of Control* 18, 217–238. URL: <http://dx.doi.org/10.3166/ejc.18.217-238>, doi:10.3166/EJC.18.217-238.
- Shokri-Manninen, F., Vain, J., Waldén, M., 2020. Formal Verification of COLREG-Based Navigation of Maritime Autonomous Systems, in: *Lecture Notes in Computer Science*, pp. 41–59.
- Smogeli, O., 2015. Managing DP System Software - A Life-cycle Perspective. *IFAC-PapersOnLine* 48, 324–334.
- Smogeli, O., Ludvigsen, K.B., Jamt, L., Vik, B., Nordahl, H., Kyllingstad, L.T., Yum, K.K., Zhang, H., 2020. Open Simulation Platform – An Open-Source Project for Maritime System Co-Simulation, in: *19th Conference on Computer and IT Applications in the Maritime Industries*, pp. 239–253.
- Smogeli, O., Skogdalen, J.E., 2011. Third party HIL testing of safety critical control system software on ships and rigs, in: *Offshore Technology Conference, One Petro*. pp. 839–845.
- Thyri, E.H., Breivik, M., Lekkas, A.M., 2020. A Path-Velocity Decomposition Approach to Collision Avoidance for Autonomous Passenger Ferries in Confined Waters. *IFAC-PapersOnLine* 53, 14628–14635. URL: <https://doi.org/10.1016/j.ifacol.2020.12.1472>, doi:10.1016/j.ifacol.2020.12.1472.
- Torben, T.R., Glomsrud, J.A., Pedersen, T.A., Utne, I.B., Sørensen, A.J., 2022a. Automatic Simulation-based Testing of Autonomous Ships using Gaussian Processes and Temporal Logic. *Journal of Risk and Reliability*, 1–21.
- Torben, T.R., Smogeli, O., Utne, I.B., Sørensen, A.J., 2022b. On Formal Methods for Design and Verification of Maritime Autonomous Surface Ships, in: *Proceedings of the 7th World Maritime Technology Conference, Copenhagen*. pp. 251–262.
- Utne, I.B., Rokseth, B., Sørensen, A.J., Vinnem, J.E., 2020. Towards supervisory risk control of autonomous ships. *Reliability Engineering and System Safety* 196, 106757. URL: <https://doi.org/10.1016/j.res.2019.106757>, doi:10.1016/j.res.2019.106757.
- Vasstein, K., 2021. A high fidelity digital twin framework for testing exteroceptive perception of autonomous vessels.
- Vasstein, K., Brekke, E.F., Mester, R., Eide, E., 2020. Autoferry Gemini: A real-time simulation platform for electromagnetic radiation sensors on autonomous ships. *IOP Conference Series: Materials Science and Engineering* 929. doi:10.1088/1757-899X/929/1/012032.

Towards Contract-based Verification for Autonomous Vessels

Woerner, K., Benjamin, M.R., Novitzky, M., Leonard, J.J., 2019. Quantifying protocol evaluation for autonomous collision avoidance: Toward establishing COLREGS compliance metrics. *Autonomous Robots* 43, 967–991. doi:10.1007/s10514-018-9765-y.

Paper C:

Resetting observer design for linear time-varying systems with application to dynamic positioning of marine surface vessels

Tobias Rye Torben, Andrew R. Teel, Øivind K. Kjerstad, Emilie H. T. Wittemann and Roger Skjetne

Provisionally accepted for publication in IEEE Transactions on Control Systems Technology

A resetting observer for linear time-varying systems with application to dynamic positioning of marine surface vessels

Tobias R. Torben¹, Andrew R. Teel², Øivind K. Kjerstad³, Emilie H. T. Wittemann⁴ and Roger Skjetne¹

Abstract—This paper presents a resetting observer for linear time-varying (LTV) systems. The motivation for the observer is better handling of unmodeled dynamics and reactivity to external disturbances without compromising steady-state performance. A reset is triggered if the output estimation error exceeds predefined bounds. The proposed observer uses a finite-time observer (FTO) approach to calculate corrected state estimates after a reset is triggered. The FTO equations are derived for LTV systems, and a method for calculating the state transition matrices online is presented. The observer equations are formulated in a hybrid dynamical systems framework, and sufficient conditions for uniform global pre-asymptotic stability are given. The method is applied to observer design for dynamic positioning of marine surface vessels. Numerical simulations as well as model scale experiments of this application show promising results, with improved transient performance compared to state-of-the-art observers.

Index Terms—Observer design, Hybrid dynamical systems, Linear time-varying systems, Finite-time observers, Marine surface vessels, Dynamic positioning.

I. INTRODUCTION

Observers play a vital role in control systems. The main objective of an observer is to estimate the states of a system based on partial and uncertain measurements. Also, an observer may have a signal processing role, where it filters noise and unwanted frequency components before the signal enters the control loop. Dynamic Positioning (DP) of a marine vessel is the process of automatic position and heading control by dynamically controlling the thrusters [1]. The model-based nonlinear passive observer (NPO) of [2] is state-of-the-art in industrial DP systems. The NPO takes only uncertain pose measurements as input and estimates pose, velocity, and a bias load. A challenge for the NPO is to handle unmodeled dynamics and external time-varying disturbances in a reactive manner, without using too high injection gains causing measurement noise to be amplified and unwanted oscillations to occur in the state estimates. Addressing this challenge has the potential to enhance the transient behaviour of the control system and reduce fuel consumption.

Several applications today call for DP systems able to handle rapidly varying disturbances and transients. Examples include DP

This work was supported by the Research Council of Norway through the Centres of Excellence funding scheme, project number 223254 - NTNU AMOS, the KPN ORCAS project, project number 280655 and the Air Force Office of Scientific Research grant FA9550-21-1-0452.

¹Centre for Autonomous Marine Operations (AMOS), Department of Marine Technology, Norwegian University of Science and Technology (NTNU), Otto Nielsens vei 10, 7052 Trondheim, Norway (tobias.torben, roger.skjetne@ntnu.no)

²Institute of Electrical and Electronics Engineers, University of California, Santa Barbara (UCSB), Santa Barbara, CA 93106, USA teel@ucsb.edu

³Department of Ocean Operations and Civil Engineering, Norwegian University of Science and Technology (NTNU), Larsgårdsvegen 2, 6002, Alesund, Norway oivind.k.kjerstad@ntnu.no

⁴TechnipFMC, Philip Pedersens vei 7, 1366 Lysaker, Norway emiliewittemann@gmail.com

in ice, anchor handling operations, subsea pipe laying, automatic docking and DP while interacting with other fixed or floating structures. In recent years, several observers and controllers that improve the transient DP performance have been proposed. An effective approach is to use velocity measurements in the observer. However, high-quality velocity measurements are usually not available from GNSS receivers at low speeds, which is the primary speed domain for most DP operations. Utilizing acceleration measurements from an inertial measurement unit (IMU) is another effective approach, this has been proposed for use in DP system by GNSS aided inertial navigation [3], acceleration feedforward [4], and a hybrid observer switching between a model-based and inertial observer [5]. Other proposed approaches are implemented purely in software, and thus avoid the installation of expensive additional sensors, such as NPO with time-varying observer gains [6], and the resetting observer of [7]. The latter has inspired our approach, where the idea is to reset the state estimates if the output estimation error exceeds a predefined bound. The concept of *finite-time observers* (FTOs) appears to be a promising candidate for calculating the estimated state after a reset in a resetting observer. The FTO concept is that for an observable linear system, two Luenberger observers can be designed. By comparing the outputs of these observers, the exact system state can be calculated. FTOs first appeared in the literature in [8]. There, a continuous-time observer for a linear time-invariant (LTI) systems was developed using time delays. Later, [9] designed an FTO for LTI systems that corrects the state estimate at a predetermined time after start-up.

In this paper, we propose a hybrid observer design which combines the idea of the resetting mechanism of [7], where a reset is triggered if the estimation error exceeds predefined bounds, with an FTO approach for calculating the corrected the state estimates after a reset. The FTO concept is extended to cover linear time-varying (LTV) systems, and an observer for generic LTV systems is developed. We then show how this observer can be applied for DP in a case study including simulations, a sensitivity analysis, and model scale experiments.

The paper is outlined as follows. First, some mathematical preliminaries on linear time-varying systems and hybrid dynamical systems are given in Section II. In Section III, the novel observer design is developed and its stability is analyzed using hybrid dynamical systems theory. In Section IV the resetting observer is applied to a DP system and results from numerical simulations and model scale experiments are presented and analyzed. Some technical aspects of the approach are discussed in Section V with suggestion for further work, before concluding remarks are given in Section VI.

II. PRELIMINARIES

A. Notation

In this paper, $\|\cdot\|$ denotes the Euclidean vector norm (2-norm), and $|\cdot|$ denotes the scalar absolute value. $\|x\|_{\mathcal{A}}$ denotes the distance from the vector x to the set \mathcal{A} , that is, $\|x\|_{\mathcal{A}} := \inf_{y \in \mathcal{A}} \|x - y\|$. Set-valued

mappings are denoted by double arrows, for instance, $M : A \rightrightarrows B$ denotes a mapping M which maps values in A to subsets of B . The domain of a mapping is denoted $\text{dom}(\cdot)$. ∇f denotes the gradient vector of the scalar function f . Dot products are denoted by bracket notation: $\langle \cdot, \cdot \rangle$. The Cartesian product of A and B is denoted $A \times B$. $\lfloor \cdot \rfloor$ is the floor operator. $\kappa(\cdot)$ denotes the 2-norm condition number for inversion. x^+ denotes the value of x after a discrete jump.

B. Linear time-varying systems

We consider LTV systems of the form

$$\dot{z} = A(t)z + B(t)u(t) + \Lambda(t)d(t) \quad (1)$$

$$y = C(t)z \quad (2)$$

where for each $t \geq 0$, the state $z(t) \in \mathbb{R}^n$, the output $y(t) \in \mathbb{R}^p$, the input $u(t) \in \mathbb{R}^m$, and the disturbance $d(t) \in \mathbb{R}^q$. We assume $u(\cdot)$, $d(\cdot)$, and the matrices $A(\cdot)$, $B(\cdot)$, $C(\cdot)$, and $\Lambda(\cdot)$ are piecewise continuous and bounded functions. Under these conditions, a unique solution to (1)-(2) exists and is defined for all time [10].

We introduce the notion of exponential stability and present an important stability result for LTV systems, which will be used in the stability analysis in Section III-F. The following theorem guarantees the existence of a quadratic, time-varying Lyapunov function for uniformly globally exponentially stable (UGES) LTV systems.

Theorem 1 (Existence of a quadratic Lyapunov function [11]):

Let $x = 0$ be the exponentially stable equilibrium point of $\dot{x} = A(t)x$, i.e., there exist $k > 0$ and $\lambda > 0$ such that $\|x(t)\| \leq k\|x(t_0)\| \exp(-\lambda(t - t_0))$, $\forall t \geq t_0 \geq 0$. Suppose also that $A(\cdot)$ is continuous and bounded. Let $Q(\cdot)$ be a continuous, bounded, symmetric, and uniformly positive definite matrix. Then, there exists a continuously differentiable, bounded, symmetric, and uniformly positive definite matrix $P(\cdot)$ that satisfies

$$-\dot{P}(t) = P(t)A(t) + A(t)^\top P(t) + Q(t). \quad (3)$$

Hence, $V(x, t) = x^\top P(t)x$ is a Lyapunov function for the system for which there exist positive constants k_1 , k_2 , and k_3 such that

$$k_1\|x\|^2 \leq V(x, t) \leq k_2\|x\|^2 \quad \forall x \in \mathbb{R}^n \text{ and } t \geq 0 \quad (4)$$

$$\begin{aligned} \dot{V}(x, t) &= \frac{\partial V(x, t)}{\partial t} + \frac{\partial V(x, t)}{\partial x} A(t) \\ &\leq -k_3\|x\|^2 \quad \forall x \in \mathbb{R}^n \text{ and } t \geq 0. \end{aligned} \quad (5)$$

C. Hybrid dynamical systems

To formulate the resetting observer equations and do a formal analysis, the *hybrid dynamical systems* framework of [12] is used. Only the main concepts and the results needed in our analysis are presented here. The reader is referred to [12], [13], and references therein, for further details.

In this framework, the solution to a hybrid system is denoted $x(t, j)$, where $t \in \mathbb{R}_{\geq 0}$ is continuous time and $j \in \mathbb{N}$ is discrete time. The solutions are defined over *hybrid time domains*, formally defined in [12]. A hybrid dynamical system, $\mathcal{H} = (\mathcal{C}, F, \mathcal{D}, G)$, is modeled as a *constrained differential inclusion* and a *constrained difference inclusion*:

$$x \in \mathcal{C} \quad \dot{x} \in F(x) \quad (6a)$$

$$x \in \mathcal{D} \quad x^+ \in G(x). \quad (6b)$$

When the state $x(t, j)$ is in the *flow set* \mathcal{C} , it evolves continuously (flows) according to the *differential inclusion* $\dot{x}(t, j) \in F(x(t, j))$.

When $x(t, j)$ is in the *jump set* \mathcal{D} , it evolves discretely (jumps) according to the *difference inclusion* $x(t, j+1) \in G(x(t, j))$.

A special class of hybrid systems are *well-posed hybrid systems*. For these systems there exists an extensive toolbox of stability and robustness results. Sufficient conditions for $\mathcal{H} = (\mathcal{C}, F, \mathcal{D}, G)$ to be well-posed are provided by the *Hybrid Basic Conditions* ([12], Assumption 6.5):

- \mathcal{C} and \mathcal{D} are closed subsets of \mathbb{R}^n .
- $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is outer semicontinuous and locally bounded [14] relative to \mathcal{C} , $\mathcal{C} \subset \text{dom}(F)$, and $F(x)$ is a convex set for every $x \in \mathcal{C}$.
- $G : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is outer semicontinuous and locally bounded relative to \mathcal{D} , and $\mathcal{D} \subset \text{dom}(G)$.

Next, a hybrid Lyapunov theorem, which gives sufficient conditions for uniform global pre-asymptotic stability (UGpAS) of a closed set, is stated. The theorem is a relaxed version of [12] Theorem 3.18, where it allows non-decrease in the Lyapunov function after a jump if the duration of flow between each jump is sufficiently large.

Theorem 2 (Sufficient conditions; persistent flowing [12]): Let $\mathcal{H} = (\mathcal{C}, F, \mathcal{D}, G)$ be a hybrid system and let $\mathcal{A} \subset \mathbb{R}^n$ be closed. Suppose V is a Lyapunov function candidate for \mathcal{H} and there exist $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ and a continuous $\rho \in \mathcal{PD}$ such that

$$\alpha_1(\|x\|_{\mathcal{A}}) \leq V(x) \leq \alpha_2(\|x\|_{\mathcal{A}}) \quad \forall x \in \mathcal{C} \cup \mathcal{D} \cup G(\mathcal{D}) \quad (7a)$$

$$\langle \nabla V(x), f \rangle \leq -\rho(\|x\|_{\mathcal{A}}) \quad \forall x \in \mathcal{C}, f \in F(x) \quad (7b)$$

$$V(g) - V(x) \leq 0 \quad \forall x \in \mathcal{D}, g \in G(x). \quad (7c)$$

If, for each $r > 0$, there exists $\gamma_r \in \mathcal{K}_\infty$, $N_r \geq 0$ such that for every solution ϕ to \mathcal{H} , $\|\phi(0, 0)\|_{\mathcal{A}} \in (0, r]$, $(t, j) \in \text{dom}(\phi)$, $t + j \geq T$ imply $t \geq \gamma_r - N_r$, then \mathcal{A} is uniformly globally pre-asymptotically stable for \mathcal{H} .

The term of *pre-asymptotic* as opposed to asymptotic stability allows the possibility of a maximal solution that is not complete. For systems satisfying the hybrid basic conditions, UGpAS of a compact set also implies robust UGpAS. This ensures that vanishing perturbations do not dramatically change the behaviour of the solutions. This is an important property in a practical implementation faced with measurement noise, numerical errors, and unmodeled disturbances.

III. RESETTING OBSERVER DESIGN

A. Problem statement

Consider an LTV system of the form given in (1)-(2). The objective of this paper is to calculate a state estimate \hat{z} , knowing the input $u(t)$, and given measurements of the output $y(t)$. It is assumed nominally that the external disturbance $d(t) = 0$, $\forall t \geq 0$. This assumption may seem contradictory, as a key motivation for the observer design is increased reactivity to unmodeled disturbances. However, the following developments will show that the unmodeled disturbances can be added as internal states which are effectively estimated by the observer. Furthermore, it can be assumed that these states are constant in the model used by the observer, thereby removing the external excitation $d(t)$. The model used in the observer design is thus reduced to

$$\dot{z} = A(t)z + B(t)u \quad (8)$$

$$y = C(t)z. \quad (9)$$

We first derive how an exact state estimate for an LTV system can be obtained from two Luenberger observers. Then we propose how to use this result in a hybrid resetting observer.

B. FTO equations for LTV systems

Consider two Luenberger observers for (8)-(9) with state estimates $z_i, i \in \{1, 2\}$, and dynamics

$$\dot{z}_i = A(t)z_i + B(t)u + L_i(t)(y - C(t)z_i), \quad (10)$$

where $L_i(t) \in \mathbb{R}^{n \times p}$ is a piecewise continuous and bounded injection gain matrix. Define $A_i(t) := A(t) - L_i(t)C(t)$ and the error variables $e_i := z - z_i$, and let $L_i(t)$ be chosen such that the origin is uniformly globally exponentially stable (UGES) for the error dynamics

$$\dot{e}_i = \dot{z} - \dot{z}_i = A_i(t)e_i \quad (11)$$

for $i \in \{1, 2\}$. The solutions for these systems can be expressed in terms of the state transition matrix Φ_i [10], according to

$$e_i(t) = \Phi_i(t, t_0)e(t_0). \quad (12)$$

We seek to calculate the true system state, z , at times $t_{k+1} > t_k > \dots > t_0 \geq 0$ to enable the observer to reset the state estimates to z . At the start of each interval $[t_k, t_{k+1}]$, the two observers are initialized with the same state estimates, that is, $z_1(t_k) = z_2(t_k)$. The initial estimation error, $e(t_k)$ is thus equal for both observers, which implies that (12) can be used to set up two vectorial equations with two unknowns, $e(t_k)$ and $z(t_{k+1})$,

$$\Phi_1(t_{k+1}, t_k)e(t_k) = z(t_{k+1}) - z_1(t_{k+1}) \quad (13)$$

$$\Phi_2(t_{k+1}, t_k)e(t_k) = z(t_{k+1}) - z_2(t_{k+1}). \quad (14)$$

Solving (13) for $e(t_k)$ yields

$$e(t_k) = \Phi_1^{-1}(t_{k+1}, t_k)(z(t_{k+1}) - z_1(t_{k+1})). \quad (15)$$

Inserting this into (14) and solving for the true system state, $z(t_{k+1})$ then yields

$$z(t_{k+1}) = (I - \Phi_2(t_k, t_{k+1})\Phi_1^{-1}(t_k, t_{k+1}))^{-1} \times \begin{bmatrix} -\Phi_2(t_k, t_{k+1})\Phi_1^{-1}(t_k, t_{k+1}) & I \end{bmatrix} \begin{bmatrix} z_1(t_{k+1}) \\ z_2(t_{k+1}) \end{bmatrix}. \quad (16)$$

Hence, if $\Phi_1(t_k, t_{k+1})$ and $I - \Phi_2(t_k, t_{k+1})\Phi_1^{-1}(t_k, t_{k+1})$ are invertible matrices, the true system state can be calculated from the state estimates $z_1(t_{k+1})$ and $z_2(t_{k+1})$. The calculated value of the true system state will be used to update the state estimates after a reset.

C. Calculating the state transition matrices

The previous section shows that we need to know the value of the state transition matrices, $\Phi_1(t_{k+1}, t_k)$ and $\Phi_2(t_{k+1}, t_k)$, to calculate the true system state at a reset. For a generic LTV system, a closed-form expression of the state transition matrix rarely exists. Also, if the time-dependence is driven by an external signal, the signal may not be known in advance. Here, we show how the state transition matrix can be numerically calculated online in an observer.

Consider a generic LTV system $\dot{x} = F(t)x$ which satisfies the conditions for existence and uniqueness given in Section II-B. Its solution is given by

$$x(t) = \Phi(t, t_0)x(t_0), \quad t \geq t_0 \geq 0, \quad (17)$$

where $\Phi(t, t_0)$ is the state transition matrix. It is trivial to show that $\Phi(t, t_0)$ is governed by the matrix differential equation

$$\dot{\Phi}(t, t_0) = F(t)\Phi(t, t_0). \quad (18)$$

Also, $\Phi(t_0, t_0) = I$, where $I \in \mathbb{R}^{n \times n}$ is the identity matrix. Therefore, the value of $\Phi(t, t_0)$ can be approximated online by numerically integrating (18) with initial condition $\Phi(t_0, t_0) = I$.

D. Design considerations

A reset is triggered if the output estimation error exceeds predefined bounds. Let the error bounds be given by $\epsilon \in \mathbb{R}_{>0}^p$. A jump is triggered if $|(y - C(t)z_1)_i| \geq \epsilon_i$ for some $i \in \{1, 2, \dots, p\}$. The error bounds should be chosen such that measurement noise do not trigger a jump.

The state estimate z_1 is used as the output of the observer. The variable z_2 is included to only accommodate an FTO state reset. The matrix $L_1(t)$ should therefore be tuned in the normal relaxed manner, to avoid measurement noise propagating into the state estimates. The matrix $L_2(i)$ should be tuned more aggressively, meaning that higher gains are used on the measurement innovation. Because a non-aggressive observer is used during steady-state conditions, and a jump is triggered only in the transient of a disturbance, this design gives the observer a ‘‘low gain - high reactivity’’ property, which is our aim.

State resets need to be separated by some dwell time in order for the FTO mechanism to robustly calculate corrected state estimates. The manifestation of this is that the condition number of $(I - \Phi_2(t_k, t_{k+1})\Phi_1^{-1}(t_k, t_{k+1}))$ will grow large as $t_{k+1} - t_k \rightarrow 0$, which gives numerical issues when calculating its inverse. On the other hand, the integral of the state transition matrices of (18) should be reset frequently to avoid modelling errors and disturbances causing drift in the state transition matrices resulting in inaccurate state resets. To control the timing of the jumps, we propose to always reset the state transition matrix integrals after a constant time interval δ . That is, they are reset at times $t_{k+1} > t_k > \dots > t_0 \geq 0$, where $t_{k+1} = t_k + \delta$. The state transition matrices are reset to the identity matrix and z_2 is reset to the value of z_1 , such that the two observers are initialized with the same estimation error before the next interval. A state reset in the output estimate, z_1 , is triggered only if $|(y - C(t)z_1)_i| \geq \epsilon_i$ for some $i \in \{1, 2, \dots, p\}$. Otherwise, z_1 is kept unchanged after the reset. Also, because conditions that ensure non-singularity of $(I - \Phi_2(t_k, t_{k+1})\Phi_1^{-1}(t_k, t_{k+1}))$ for all times are not established for generic LTV systems, we propose to jump the output state estimate, z_1 , only if the condition number of $(I - \Phi_2(t_k, t_{k+1})\Phi_1^{-1}(t_k, t_{k+1}))$ is sufficiently low. This is discussed further in Section V.

As introduced in Section III-A, if the system is expected to be subject to severe unmodeled disturbances, it may be beneficial to add the disturbances as states to be estimated by the observer. Furthermore, it can be assumed that these states are constant in the model used by the observer. Every δ seconds, the resetting mechanism will check if $|(y - C(t)z_1)_i| \geq \epsilon_i$ for some $i \in \{1, 2, \dots, p\}$. If this is the case, the system is likely subject to an unmodeled disturbance and a reset of the state estimates will occur using the FTO approach. Equation (16) will then calculate the correct magnitude for a constant disturbance acting over the previous δ seconds and update the disturbance state estimates accordingly.

Finally, our experience has shown that resetting to the true system state may cause overshooting behaviour after a reset due to measurement noise and disturbances causing inaccuracies in the FTO

estimates. To address this challenge, we propose to add a tunable linear interpolation to the jump map. That is, instead of jumping directly to the z value computed by (16), the system jumps to $kz + (1-k)z_1$, where $k \in [0, 1]$ is tunable scalar.

E. Hybrid observer equations

We are now ready to state the hybrid observer equations for the resetting observer using the hybrid dynamical systems framework introduced in Section II-C.

The state of the hybrid system is defined as

$$x = (z, z_1, z_2, \Phi_1, \Phi_2, \zeta, \tau) \quad (19)$$

$$\in \mathbb{K} \times \mathbb{R}^n \times \mathbb{R}^n \times M \times M \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0},$$

where z is the true system state, which is assumed to live in a compact set $\mathbb{K} \subset \mathbb{R}^n$, z_1 and z_2 are the Luenberger state estimates, and Φ_1 and Φ_2 are the state transition matrices. The latter are governed by (18), and will thus have no finite escape times. Also, since they are periodically reset to identity, these matrices will live in a compact set $M \subset \mathbb{R}^{n \times n}$. The variable ζ is a scalar timer for the resets, and τ is the time variable.

Following the developments of Section III, the flow map is expressed as the set-valued mapping

$$\dot{x} \in F(x) := \bigcup_{u \in \mathbb{U}} f(x, u), \quad (20)$$

where \mathbb{U} is a compact subset of \mathbb{R}^m and

$$f(x, u) := \begin{bmatrix} A(\tau)z + B(\tau)u \\ A_1(\tau)z_1 + B(\tau)u + (A(\tau) - A_1(\tau))z \\ A_2(\tau)z_2 + B(\tau)u + (A(\tau) - A_2(\tau))z \\ A_1(\tau)\Phi_1 \\ A_2(\tau)\Phi_2 \\ 1 \\ 1 \end{bmatrix}. \quad (21)$$

The jump map is defined as

$$x^+ = g(x) := \begin{bmatrix} z \\ h(x) \\ h(x) \\ I \\ I \\ 0 \\ \tau \end{bmatrix}, \quad (22)$$

where

$$h(x) := \begin{cases} k\Psi(x) \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + (1-k)z_1; & \text{if } |Cz_1 - y|_i \geq \epsilon_i, \\ & \text{for some } i \in \{1, 2, \dots, p\} \text{ and} \\ & \kappa \left(I - \Phi_2 \Phi_1^{-1} \right) \leq c_{max} \\ z_1; & \text{otherwise} \end{cases} \quad (23)$$

for $k \in [0, 1]$ and

$$\Psi(x) := (I - \Phi_2 \Phi_1^{-1})^{-1} \begin{bmatrix} -\Phi_2 \Phi_1^{-1} & I \end{bmatrix}.$$

The threshold $c_{max} \in \mathbb{R}_{\geq 1}$ is the highest value of the condition number of $I - \Phi_2 \Phi_1^{-1}$ for which a reset using Ψ can occur.

The flow set is defined as

$$\mathcal{C} := \mathbb{K} \times \mathbb{R}^n \times \mathbb{R}^n \times M \times M \times [0, \delta] \times \mathbb{R}_{\geq 0}, \quad (24)$$

and the jump set is defined as

$$\mathcal{D} := \mathbb{K} \times \mathbb{R}^n \times \mathbb{R}^n \times M \times M \times \{\delta\} \times \mathbb{R}_{\geq 0}. \quad (25)$$

Together, this completely defines a hybrid system $\mathcal{H} = (\mathcal{C}, F, \mathcal{D}, g)$. Stability of the observer is addressed next.

F. Formal stability analysis

The resetting observer (20)-(25) contains the conditional statement $h(x)$ in the jump map. This results in a jump map that is not outer semicontinuous, and therefore a hybrid system that does not satisfy the hybrid basic conditions. However, for the purpose of the formal analysis, we can construct a generalized hybrid system $\mathcal{H}' = (\mathcal{C}, F, \mathcal{D}, G)$ by replacing g with its outer semicontinuous hull G [14]. This involves replacing the conditional statement h with a set-valued mapping H which contains both values at the boundary of the conditional statement:

$$H(x) := \begin{cases} k\Psi(x) \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + (1-k)z_1; & \text{if } |Cz_1 - y|_i > \epsilon_i, \\ & \text{for some } i \in \{1, 2, \dots, p\} \text{ and} \\ & \kappa \left(I - \Phi_2 \Phi_1^{-1} \right) < c_{max} \\ z_1; & \text{if } |Cz_1 - y|_i < \epsilon_i, \forall i \in \{1, 2, \dots, p\} \text{ or} \\ & \kappa \left(I - \Phi_2 \Phi_1^{-1} \right) > c_{max} \\ \{z_1\} \cup \left\{ k\Psi(x) \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + (1-k)z_1 \right\}; & \text{otherwise} \end{cases} \quad (26)$$

All solutions of \mathcal{H} are contained in the set of solutions of \mathcal{H}' , and UGpAS of a set \mathcal{A} for \mathcal{H} therefore follows from UGpAS of the set \mathcal{A} for \mathcal{H}' . We begin by showing that \mathcal{H}' satisfies the hybrid basic conditions.

First we note that \mathcal{C} and \mathcal{D} are closed sets. F is locally bounded, outer semicontinuous and has convex values for every $x \in \mathcal{C}$. G is locally bounded and outer semicontinuous relative to \mathcal{D} . Continuity of $\Psi(\cdot)$ follows from the fact that the matrix inverse is a continuous function for non-singular matrices, and that continuity is conserved through products, sums, and compositions with other continuous functions. This shows that the system of (20)-(22) satisfies all the hybrid basic conditions and, therefore, constitutes a well-posed hybrid system. Having this property means that if we can prove that a set is UGpAS for \mathcal{H}' , then it is also robustly UGpAS. The following stability result gives sufficient conditions for establishing UGpAS.

Theorem 3: Let $u(\cdot)$, $A(\cdot)$, $B(\cdot)$, $C(\cdot)$, $L_1(\cdot)$, and $L_2(\cdot)$ be piecewise continuous and bounded functions, and L_i be chosen such that the origin is UGES for $\dot{e}_i = A_i(t)e_i$ for $i \in 1, 2$. Furthermore, let $\delta > 0$ and $k \in [0, 1]$. Then, the set $\mathcal{A} := \{(z, z_1) \in \mathbb{K} \times \mathbb{R}^n : z = z_1\} \times \mathbb{R}^n \times M \times M \times [0, \delta] \times \mathbb{R}_{\geq 0}$ is UGpAS for the hybrid system \mathcal{H}' .

Proof: First we note that $\|x\|_{\mathcal{A}} = \|z - z_1\| = \|e_1\|$ since z_2 , Φ_1 , Φ_2 , ζ , and τ are always in their respective subsets of \mathcal{A} by construction. We also note that the time-dependence of the LTV system has been replaced by the time state, τ , in the hybrid system. We can thus analyze stability of a non-compact set for a time-invariant hybrid system using Theorem 2. Forward completeness is guaranteed by the conditions on u , A , B , C , L_1 , and L_2 .

As the error dynamics of e_1 are governed by the UGES LTV system $\frac{de_1}{d\tau} = A_1(\tau)e_1$, Theorem 1 implies that there exist $Q(\cdot)$, $P(\cdot)$, and $V_0(e_1, \tau) := e_1^\top P(\tau)e_1$ that satisfy the conditions of Theorem 2. It follows that $V(x) := V_0(e_1, \tau)$ is a Lyapunov function candidate for \mathcal{H}' that satisfies conditions (7a) and (7b) of Theorem 2.

Next, we show that \mathcal{H}' also satisfies (7c), that is, $V(x)$ does not increase after jumps. We have that

$$e_1^+ = e_1 \quad (27)$$

or

$$e_1^+ = z^+ - z_1^+ = z - (k\Psi(x) \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + (1-k)z_1). \quad (28)$$

In the first case, $V(x^+) - V(x) = 0$, and (7c) is satisfied. In the second case, we have that $\Psi(x) \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = z$, as shown in Section III-B. It follows that

$$e_1^+ = z - (z_1 + k(z - z_1)) = (1-k)e_1. \quad (29)$$

The value of the Lyapunov function after a jump then becomes

$$\begin{aligned} V(x^+) &= (e_1^+)^{\top} P(\tau)e_1^+ = ((1-k)e_1)^{\top} P(\tau)((1-k)e_1) \\ &= (1-k)^2 e_1^{\top} P(\tau)e_1 = (1-k)^2 V(x) \leq V(x) \end{aligned} \quad (30)$$

since $k \in [0, 1]$. Hence, condition (7c) is also satisfied.

What remains to show is that the duration of flow between each jump is sufficiently large to compensate for the potential non-decrease in the Lyapunov function after jumps. Since a jump always occurs every $\delta \geq 0$ time units we have that

$$j = \lfloor \frac{t}{\delta} \rfloor \leq \frac{t}{\delta} \implies t \geq \delta j. \quad (31)$$

Adding δt to both sides of the inequality gives

$$\delta t + t \geq \delta j + \delta t \implies t(1 + \delta) \geq \delta(t + j). \quad (32)$$

Rearranging terms, we arrive at a \mathcal{K}_∞ relation between t and $t + j$

$$t \geq \frac{\delta}{1 + \delta}(t + j). \quad (33)$$

Choosing $\gamma_r(T) = \frac{\delta}{1 + \delta}T \in \mathcal{K}_\infty$ and $N_r = 0$, the conditions of Theorem 2 are satisfied for every $r > 0$.

Together, this shows that we can conclude UGpAS of \mathcal{A} for the hybrid system \mathcal{H}' . In fact, since V is exponentially decreasing when it evolve continuously and non-increasing when it resets, and all solutions are forward complete, the stability result of this theorem can be strengthened to UGES. ■

IV. CASE STUDY: DYNAMIC POSITIONING

The resetting observer of (20)-(25) applies to generic observable LTV systems. In this section we show how it can be applied for a DP system.

A. Mathematical modelling and observer design

The standard control design model for the low-frequency motion of a marine surface is defined as

$$\dot{\eta} = R(\psi)\nu \quad (34a)$$

$$\dot{b} = d(t) \quad (34b)$$

$$M\dot{\nu} + D\nu = \tau + R(\psi)^\top b \quad (34c)$$

$$y = \eta, \quad (34d)$$

where $\eta \in \mathbb{R}^3$ is the position and heading, $\nu \in \mathbb{R}^3$ is the body frame velocity and turn rate, $b \in \mathbb{R}^3$ is a bias load, $d(t) \in \mathbb{R}^3$ is the disturbance, and $\tau \in \mathbb{R}^3$ compiles the body frame thruster forces. $R(\psi) \in \mathbb{R}^{3 \times 3}$ is a three degree of freedom rotation matrix, $M \in \mathbb{R}^{3 \times 3}$ is the mass matrix, including added mass, and $D \in \mathbb{R}^{3 \times 3}$ is the linear damping matrix.

The system of (34) is a highly simplified control design model which attempts to capture the main dynamics of the complex hydrodynamic interactions between the vessel, thrusters, and water. In this model, the bias state is used as an internal state to estimate unmodeled loads, and in the observer model it will be assumed that $d(t) = 0$, as suggested in Section III-D.

Equation (34) is a nonlinear model due to the rotation matrices. However, since the heading is measured by a gyrocompass within the compact interval of $[-\pi, \pi]$ and the heading rate (its derivative) is bounded due to vessel damping and limited thruster forces, we can use the heading measurement directly in the rotation matrix as an external time signal and assuming $R(t) := R(\psi(t))$ to be a time-varying matrix [15]. The nonlinear system of (34) can then be written in LTV form as

$$z = [\eta^\top, b^\top, \nu^\top]^\top \in \mathbb{R}^9, \quad u = \tau \in \mathbb{R}^3 \quad (35a)$$

$$\dot{z} = A(t)z + Bu \quad (35b)$$

$$y = Cz, \quad (35c)$$

$$A(t) = \begin{bmatrix} 0_{3 \times 3} & 0_{3 \times 3} & R(t) \\ 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & M^{-1}R(t)^\top & -M^{-1}D \end{bmatrix} \quad (36)$$

$$B = \begin{bmatrix} 0_{3 \times 3} \\ 0_{3 \times 3} \\ M^{-1} \end{bmatrix} \quad (37)$$

$$C = [I_{3 \times 3} \quad 0_{3 \times 3} \quad 0_{3 \times 3}]. \quad (38)$$

Luenberger observers for z_1 and z_2 can then be designed as

$$\dot{z}_i = A(t)z_i + Bu + L_i(t)(y - Cz_i), \quad (39)$$

where

$$L_i(t) = \begin{bmatrix} K_{1,i} \\ K_{2,i} \\ M^{-1}R^\top(t)K_{3,i} \end{bmatrix} \in \mathbb{R}^{9 \times 3} \quad (40)$$

such that the origin is UGES for $\dot{e}_i = A_i(t) := A(t) - L_i(t)C$. These Luenberger observers can now readily be used in the resetting observer of (20)-(25).

B. Simulation study

To evaluate the performance of the proposed observer design, it was tested in simulation with a high-fidelity simulation model. The model used is the *Supply Vessel* from Marine Systems Simulator [16]. The parameter values used in the simulation study are given in Table I.

Parameter	Value
Differential GNSS standard deviation	0.1m
Dual-band GNSS standard deviation	1.5m
Heading sensor standard deviation	0.3°
Sensor Gauss-Markov time constant (1/c)	1100s
Sensor sample time (T_s)	1.0s
Notch filter central frequency (ω_0)	1.0rad/s
Notch filter band width (ω_c)	0.5rad/s
Reset time (δ)	2.5s
Interpolation gain (k)	0.7
Estimation error bounds (ϵ)	[0.5m, 0.5m, 0.05rad]
Process covariance for L_1 (Q_1)	diag([50, 50, 10])
Measurement covariance for L_1 (R_1)	diag([15, 15, 1])
Process covariance for L_2 (Q_2)	$10^{-7}Q_1$
Measurement covariance for L_2 (R_2)	R_1

TABLE I

THE PARAMETER VALUES USED IN THE SIMULATION STUDY.

The measurement error of the GNSS east and north components and heading sensor are modeled as Gauss-Markov processes [17], that is,

$$v[n+1] = e^{-cT_s}v[n] + \rho[n], \quad (41)$$

where $v[n]$ is the measurement at discrete time n , $\frac{1}{c}$ is the time constant for the process, T_s is the sampling time and ρ is zero-mean Gaussian white noise with standard deviation, σ . The values of c , T_s , and σ are chosen to match the characteristics of commercially available differential and dual-band GNSS and heading sensors.

To remove wave-frequency components of the measurements, they are notch filtered before entering the observer. The notch filter used is a linear second-order filter with transfer function

$$H(s) = \frac{s^2 + \omega_0^2}{s^2 + \omega_c s + \omega_0^2}, \quad (42)$$

where ω_0 is the central frequency and ω_c is the width of the rejected band. The injection gain matrices for the Luenberger observers, L_1 and L_2 , were obtained using optimal gains from a linear Kalman filter design about zero heading. The L_1 gains were first calculated by tuning the process covariance matrix Q_1 and the measurement covariance matrix, R_1 . Then, the more aggressive L_2 gains were calculated by keeping $R_2 = R_1$ while setting Q_2 to a significantly lower value of $10^{-7}Q_1$.

1) Transient and steady state performance: As introduced in Section I, our motivation for this observer design is to achieve increased reactivity against rapidly changing disturbances without compromising the steady state performance. To investigate whether the resetting observer accomplishes this goal, the vessel was excited by an impulsive sway disturbance with magnitude 5000kN and duration 10s at $t = 50s$, followed by a long period of steady-state operation. The sea state in the simulation was governed by a JONSWAP wave spectrum [18] with significant wave height 2m and peak period 6.3s.

The results are presented in Figure 1. Note that the position plot shows the estimation error whereas the velocity and bias plots show the estimated and true values. The results marked “continuous” show state estimates from the NPO of [2], which is equivalent to the LTV Luenberger observer with innovation gain $L_1(t)$ in the resetting observer. The results show promising performance. The resetting observer gives a substantial improvement during the transient phase, without amplifying measurements noise or introducing wave frequency components to the state estimates. The vessel is subject to a severe disturbance, but this is captured well by the resetting mechanism in the bias estimate, which estimates a constant value for

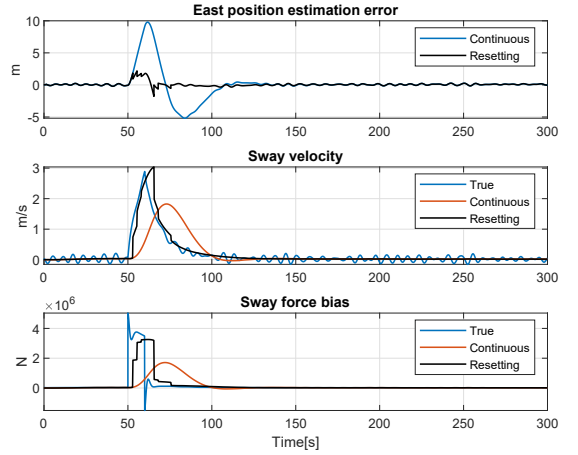


Fig. 1. The simulation results for evaluating transient and steady state performance.

the bias for each δ interval. It can be seen that there is a substantial wave-frequency component in the sway velocity. However, since this measurement is notch-filtered before entering the observer, this does not trigger unwanted jumps. There is some delay in the state estimates of the resetting observer due to the resetting time δ and the phase-lag introduced by the notch filter. When tuning the notch filter, there must be a trade-off between phase-lag and wave attenuation. This should be adjusted to the prevailing sea state to ensure that the wave-frequency component of the measurements can not trigger jumps.

2) Sensitivity analysis of observer parameters: To investigate the effect of varying the tunable parameters of the resetting mechanism, a sensitivity analysis was performed on the interpolation gain, k , reset time δ , and error bounds ϵ . To obtain an objective and quantitative measure of the observer performance in a simulation, a key performance indicator (KPI) was developed based on the estimation error. In order to combine variables with different physical units in the same KPI, all variables were non-dimensionalized using the BIS system [18]. The KPI, J_{tot} is defined by

$$J_{tot} = \int_0^T \|z_{bis}(t) - \hat{z}_{bis}(t)\| dt, \quad (43a)$$

where T is the length of the simulation. Estimated state variables, obtained from z_1 of the resetting observer, are denoted by hatted symbols. Lower values of the KPI indicate better observer performance.

A total of 100 simulations were run for each of the tuning parameters k , δ , and ϵ , where their values were varied within a predefined range. The other parameters were kept unchanged at the values given in Table I. To investigate the relationship between measurement noise and tuning parameters, the sensitivity analysis was conducted both with a differential and a dual-band GNSS model. Each simulation had a duration $T = 1000s$, where the vessel was subject to both impulsive and slowly-varying disturbances in surge, sway, and yaw followed by a longer period of steady state operation with only wave and measurement noise disturbances. The disturbances induced simultaneous heading and surge/sway motion, thereby exciting the time-varying (nonlinear) dynamics of the system.

First, the effect of the interpolation gain k is studied for values in the interval $[0, 1]$, which is the entire allowable range for k .

Using $k = 0$ is equivalent to a continuous-time NPO observer, completely disregarding the FTO estimates, while for $k = 1$ we use the FTO estimates directly after a reset. The top plot in Figure 2 shows the trend that the performance improves as k is increased, and performance is drastically improved compared to the NPO score at $k = 0$. The improvement flattens out for higher values of k . As we discussed in Section III-D, having too high values of k may lead to overshoots after a reset due to inaccuracies caused by measurement noise and unmodeled disturbances, and it is therefore more advantageous to do several more gradual jumps by using a lower value of k . The results of Figure 2 underpins this theory, as the simulations with higher measurement noise (dual-band) perform worse for high-values of k . Our choice of using $k = 0.7$ appears to find a good balance.

Next, the effect of varying the reset time δ is studied. The middle plot of Figure 2 shows the trend that the performance decreases as the reset time increases. This is as expected, since a higher reset time means that the resetting mechanism must wait longer before it can reset to a more correct state estimate. More surprisingly, the resetting observer produces excellent state estimates also for very low values of δ . As noted in Section III-D, the condition number of $(I - \Phi_2 \Phi_1^{-1})$ in the jump map grows large when δ is decreased. We have found that when δ approaches the time step size of the control system, which in this case was $0.05s$, the observer becomes unstable if we also disable the check on the condition number in the jump map. In these cases the condition number of $(I - \Phi_2 \Phi_1^{-1})$ is of the order of 10^{16} . Since we are using double precision floating point arithmetic in the simulations, the machine epsilon is $\approx 10^{-16}$, meaning that the matrix is so ill-conditioned that numerical round-off error will be amplified to the degree that the results are useless. However, we found that when δ is greater than about three times the time step size, the observer works perfectly fine even though the condition number of $(I - \Phi_2 \Phi_1^{-1})$ is of the order of 10^{10} . This indicates that the resetting observer design is highly robust against $(I - \Phi_2 \Phi_1^{-1})$ being ill-conditioned. In all other simulations $\delta = 2.5s$ is used. As Figure 2 shows, there are no significant performance gains by using a lower δ than this. With $\delta = 2.5s$ the condition number of $(I - \Phi_2 \Phi_1^{-1})$ has values around 50, giving large robustness margins before singularity becomes an issue.

Finally, the effect of varying the error bounds ϵ is investigated. In the other simulations, $\epsilon = [0.5m, 0.5m, 0.05rad]$ was used. In the sensitivity analysis, scaled versions of this is used, that is, $\epsilon = c_\epsilon [0.1m, 0.1m, 0.01rad]$, with $c_\epsilon \in [0.1, 10]$. The bottom plot in Figure 2 shows that the performance is poor for very low values of ϵ . This occurs because measurement noise constantly triggers unwanted resets. This effect is more pronounced for the simulations with higher measurement noise (dual-band). Conversely, for high values c_ϵ the performance is decreasing slightly with increasing ϵ due to lower reactivity when the error bounds are too big. The tuning used in the other simulations correspond do a c_ϵ value of 5, which appears to find a good balance based the results in Figure 2.

An important finding from the sensitivity analysis is that the observer has good performance over a wide range of values for the tunable parameters of the resetting mechanism. Based on the results of the sensitivity analysis, we recommend that k should be reduced if there is high measurement noise. The value of δ should be chosen significantly larger than the time step of the control system and can generally be chosen much larger than this without negatively impacting performance. The ϵ bounds should be set larger than the prevailing measurement noise.

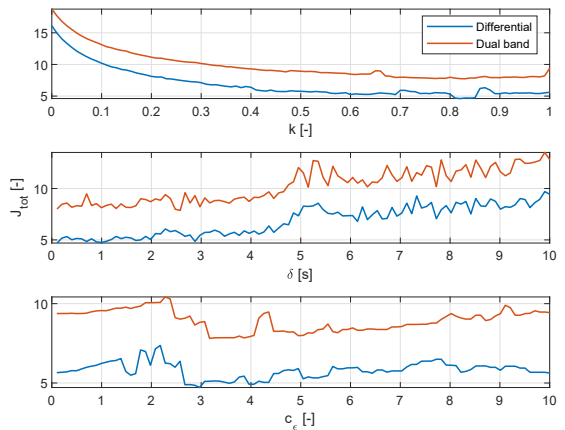


Fig. 2. Sensitivity analysis investigating effect of the tunable parameters k , δ and ϵ on the KPI J_{tot} .



Fig. 3. The Cybership III ship model in the MCLab wave tank.

C. Experimental study

To validate the results from the simulation study, the resetting observer was tested experimentally in model scale experiments. The experiments were conducted in the Marine Cybernetics Laboratory (MC-Lab) wave tank at the Norwegian University of Science and Technology (NTNU). The test vessel used was Cybership III, a 1:30 scale model of an offshore supply vessel, as shown in Figure 3. For more details on the experimental study, the reader is referred to [19]. This thesis includes more detailed experimental results and performs a comparison of the resetting observer with the NPO of [2] and the NPO with time-varying gains of [6]. The thesis compared the three observers in ten different simulated scenarios and five different experimental scenarios. The results were highly convincing of the merits of the resetting observer, showing that the resetting observer performed better than the NPO in all simulated and experimental scenarios, and it performed better than the NPO with time-varying gains in nine of ten simulated scenarios and all experimental scenarios.

In this paper we give the results of one experimental scenario which attempts to replicate the simulation in Section IV-B.1 by giving the ship model a push in the sway-direction, thereby exerting an impulse-like, unmodeled disturbance. The experiment was conducted in a moderate sea state, generated from a JONSWAP spectrum with significant waveheight $0.04m$ ($1.2m$ full-scale) and peak period $0.8s$

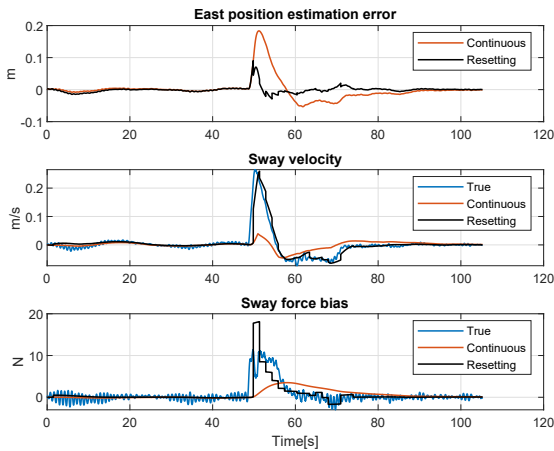


Fig. 4. Experimental results for the resetting observer.

(4.4s full-scale). The results from the experiment are shown in Figure 4. The results validate the findings in the simulation study by showing similar behavior and performance gain compared to a continuous-time observer. The results show that the peak of bias estimate from the resetting observer is higher than the estimated true bias signal. We believe this is an artifact of the filtering used in estimation of the true bias signal, which has filtered out the actual peak.

V. DISCUSSION AND FURTHER WORK

Our work does not include conditions to ensure non-singularity of $(I - \Phi_2 \Phi_1^{-1})$ for all times. [8] gives sufficient, but highly conservative conditions for non-singularity in the case of linear time-invariant systems. However, the extension of these conditions to the time-varying case is not trivial. The paper [20] proposes a workaround for this problem for uniformly observable systems LTV system by transformation to observer canonical form and separating out and cancelling the time-varying dynamics resulting in time-invariant error dynamics. The state-transition matrix is thus the matrix exponential and non-singularity $(I - \Phi_2 \Phi_1^{-1})$ can therefore be established up front by inspection. This approach is, however, not attractive for the DP application because the transformation to observer canonical form requires signals which are not typically available in DP systems. In our work we use a practical solution to avoid inverting a singular matrix by adding a check on the condition number of $(I - \Phi_2 \Phi_1^{-1})$ before doing a jump, as stated in (22). In our experience this has worked very well in practice. Our experience shows that despite the time-varying dynamics of the observer, the condition number stays practically constant throughout a simulation. The main factor influencing the condition number is the reset time δ . By running some simulations, the relationship between δ and the condition number of $(I - \Phi_2 \Phi_1^{-1})$ can be established. This relationship can be used to find a safe value for δ . Combining this knowledge with the results showing that the observer is highly robust against ill-conditioned matrices and the check of condition number in the jump map, the singularity of $(I - \Phi_2 \Phi_1^{-1})$ does not seem to be a likely problem in practice. Nevertheless, establishing a priori conditions for non-singularity would be a valuable addition to the observer design, and we leave this for future work.

VI. CONCLUSIONS

We have presented a resetting observer for LTV systems. A reset is triggered if the output estimation error exceeds predefined bounds. To calculate corrected state estimates after a reset, an FTO approach is used. The FTO equations have been derived for LTV systems. The observer equations have been formulated in a hybrid dynamical systems framework, and sufficient conditions for UGPAS have been given. The observer design has applications for DP of marine surface vessels. A case study for this application was conducted with numerical simulations and an experimental demonstration. The results showed promising results, with improved transient performance without compromising steady-state performance compared to the state-of-the-art continuous-time observer. These properties may enable DP operations in more challenging conditions, as well as better noise filtering properties due to the low injection gains. The resetting observer may also lower the requirements for model and parameter accuracy, as it more rapidly captures and corrects for model errors. The discussion highlights developing conditions that guarantee non-singularity in the resetting mechanism as a topic for future research.

REFERENCES

- [1] A. J. Sørensen, "A survey of dynamic positioning control systems," *Annual Reviews in Control*, vol. 35, no. 1, pp. 123–136, 2011.
- [2] T. I. Fossen and J. P. Strand, "Passive nonlinear observer design for ships using Lyapunov methods," *Automatica*, vol. 35, p. 3–16, 1999.
- [3] T. H. Bryne, R. H. Rogne, T. I. Fossen, and T. A. Johansen, "A virtual vertical reference concept for aided inertial navigation at the sea surface," *Control Engineering Practice*, vol. 70, no. March 2017, pp. 1–14, 2018.
- [4] Ø. K. Kjerstad and R. Skjetne, "Disturbance Rejection by Acceleration Feedforward for Marine Surface Vessels," *IEEE Access*, vol. 4, pp. 2656–2669, 2016.
- [5] A. H. Brodtkorb, S. A. Værnø, A. R. Teel, A. J. Sørensen, and R. Skjetne, "Hybrid controller concept for dynamic positioning of marine vessels with experimental results," *Automatica*, vol. 93, pp. 489–497, 2018.
- [6] S. A. Vaerno, A. H. Brodtkorb, R. Skjetne, and V. Calabro, "Time-varying model-based observer for marine surface vessels in dynamic positioning," *IEEE Access*, vol. 5, pp. 14787–14796, 2017.
- [7] Ø. K. Kjerstad, *Dynamic Positioning of Marine Vessels in Level Ice*. PhD thesis, NTNU, 2016.
- [8] R. Engel and G. Kreisselmeier, "A Continuous-Time Observer Which Converges in Finite Time," *IEEE Transactions on Automatic Control*, vol. 47, no. 7, pp. 1202–1204, 2002.
- [9] T. Raff and F. Allgöwer, "An impulsive observer that estimates the exact state of a linear continuous-time system in predetermined finite time," *2007 Mediterranean Conference on Control and Automation, MED*, no. 2, pp. 2–4, 2007.
- [10] C.-T. Chen, *Linear System Theory and Design*. Oxford University Press, third ed., 1999.
- [11] H. K. Khalil, *Nonlinear Systems*. Prentice-Hall, 2 ed., 2002.
- [12] R. Goebel, R. G. Sanfelice, and A. R. Teel, *Hybrid Dynamical Systems: Modeling, Stability, and Robustness*. Princeton University Press, 2012.
- [13] R. Goebel, R. G. Sanfelice, and A. R. Teel, "Hybrid dynamical systems," *IEEE Control Systems Magazine*, vol. 29, no. 2, pp. 28–93, 2009.
- [14] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*. 1998.
- [15] S. A. Værnø, R. Skjetne, Ø. K. Kjerstad, and V. Calabro, "Comparison of control design models and observers for dynamic positioning of surface vessels," *Control Engineering Practice*, vol. 85, no. 1999, pp. 235–245, 2019.
- [16] T. Perez, O. Smogeli, T. I. Fossen, and A. J. Sørensen, "An overview of the marine systems simulator (MSS): A simulink® toolbox for marine control systems," *Modeling, Identification and Control*, 2006.
- [17] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft, Theory and Practice*. 2012.
- [18] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, 2011.
- [19] E. H. T. Wittemann, "Master thesis: Hybrid Observers for Autonomous Surface Vessels Experiencing Varying Operational Conditions." 2021.
- [20] P. H. Menold, R. Findeisen, and F. Allgöwer, "Finite time convergent observers for linear time-varying systems," *Proceedings of the 11th Mediterranean conference on control and automation (17-078)*, no. December, pp. 5673–5678, 2013.

Paper D:

On Formal Methods for Design and Verification of Maritime Autonomous Surface Ships

**Tobias Rye Torben, Øyvind Smogeli, Ingrid B. Utne and Asgeir J.
Sørensen**

Proceedings of the 2022 World Maritime Technology Conference
Pages 253-262

On Formal Methods for Design and Verification of Maritime Autonomous Surface Ships

Tobias R. Torben¹, Øyvind Smogeli^{1,2}, Ingrid B. Utne¹ and Asgeir J. Sørensen¹

ABSTRACT

Maritime Autonomous Surface Ships (MASS) are approaching a reality, introducing a new level of complexity and criticality to maritime control systems. In this paper we investigate how Formal Methods (FMs) can be used to design and verify maritime control systems for safe and effective MASS. FMs are a family of mathematically based methods for specification and verification. We begin by giving a high-level introduction to FMs. We discuss the current practice for certification of maritime control systems and needs going towards autonomy. We give three specific examples on how FMs can be applied to meet these needs: Formal specification of COLREG, contract-based design and automation of simulation-based testing. Finally, some limitations of FMs are discussed. We conclude that FMs appear as a promising candidate to meet some of the needs going towards autonomy, and encourage further research into FMs for MASS.

KEY WORDS

Maritime Autonomous Surface Ships, Formal Methods, Verification, Specification, Assurance

INTRODUCTION

Maritime Autonomous Surface Ships (MASS) are approaching a reality, with numerous ongoing projects ranging from small research prototypes to full-scale industrial vessels. Although several degrees of autonomy exist, MASS are typically distinguished by being able to operate independently of a human operator in a non-trivial operation, requiring situational awareness and planning abilities. These characteristics have created a need for new design methodologies among MASS developers, as well as a need for new methods and processes for safety assurance among regulators (IMO 2021, NMD 2020) and classification societies (DNV 2018).

Formal Methods (FMs) are a family of mathematically based methods for specification and verification originating from theoretical computer science (Woodcock et al. 2009). FMs offer a high level of assurance and have therefore been used actively in the development and verification of critical systems in other industries, such as aerospace and railway, for several decades. With the advent of autonomous systems, FMs have been considered as a promising candidate to address some of the assurance challenges they introduce. This has resulted in active research on FMs applied to autonomous cars and aerial vehicles over the past decade (Luckcuck et al. 2019).

The maritime industry has not yet seen a significant adoption of FMs. This seems to be changing, however, as a few articles have been published during the last year. Shokri-Manninen et al. (2020) have created a formal automata-based model of single-vessel encounters and synthesized a correct-by-construction navigation strategy. Park and Kim (2020) have synthesized a correct-by-construction controller for automatic docking of marine vessels based on reachability analysis. Foster et al. (2020) present a controller for autonomous marine vessels in form of a hybrid dynamical system and use an automated theorem prover to verify some safety invariants.

This article aims to bring FMs to the attention of the maritime community by first giving a high-level introduction. Next, we review the current practice for design and verification of maritime control systems and discuss some needs going towards autonomy. We then motivate and demonstrate the use of FMs in three specific use cases to meet these needs. Finally, we discuss some of the limitations of FMs.

¹ Centre for Autonomous Marine Operations and Systems (AMOS), Department of Marine Technology, Norwegian University of Science and Technology

² Zeabuz AS

AN INTRODUCTION TO FORMAL METHODS

FMs are motivated by the common engineering expectation that mathematical analysis will improve the performance and reliability of a system. Early development of FMs dates to the 1960's, where they were first applied to the design of logic circuits and primitive computer programs.

Figure 1 shows the main steps and activities in a formal development process. We will use this figure to introduce the different FM tools and show where they fit in the development process. We also emphasize that taking on a full-fledged formal development process with all the steps in Figure 1 may be time-consuming and may not be appropriate in all cases. However, tools and methods from individual steps, such as only creating a formal specification, can still give great value to the development process. This is often referred to as *lightweight formal methods*.

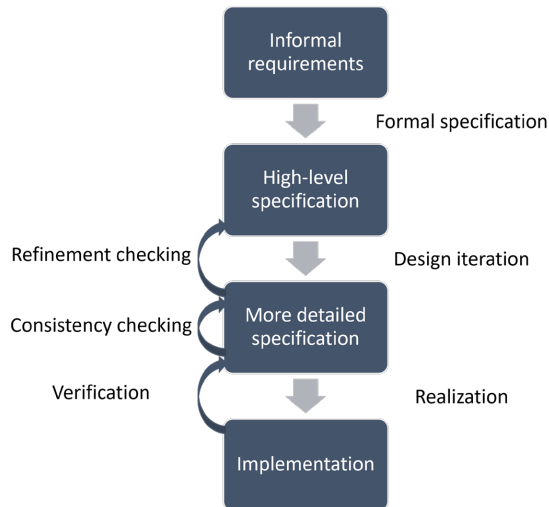


Figure 1: Steps and activities in a formal development process.

Formal specification

The first step in any development process is the requirements capture. This is often based on a text-based *concepts of operations* (CONOPS) document and produces an informal, text-based specification of high-level requirements. In a formal development process, the next step is to take the informal requirements and formulate them in a *formal specification language*.

Formal specification languages come in many forms, depending on what they are designed to describe. For instance, some are more suited for describing behaviour whereas others are better at describing structure. What they all have in common is that they have clearly defined syntax and semantics, that is, there are strict rules on valid statements, and it is clearly and unambiguously defined how the statements are to be interpreted. This not only means that they can be subject to mathematical analysis, but also that they are machine readable and, therefore, can be subject to automated reasoning by a computer.

A widely used class of specification languages are *temporal logics*, which are appropriate when specifying temporal behaviours, such as the ordering of events, timing, and avoidance of deadlocks. A temporal logic specification is written as a formula which combines logical operators, such as AND, OR, NOT and IMPLIES with temporal operators, such as ALWAYS, EVENTUALLY, NEXT and UNTIL. The original and simplest form of temporal logic is Linear Temporal Logic (LTL) (Pnueli 1977), which operates on Boolean signals in discrete time. An example of an LTL formula specifying correct behaviour of a traffic light is given below. In natural language, this formula specifies that always, if there is a red light, then it should eventually turn green after first being yellow.

ALWAYS (red IMPLIES (EVENTUALLY green AND (NOT green UNTIL yellow)))

[1]

Signal Temporal Logic (STL) (Maler and Nickovic 2004) is another type of temporal logic which can be used to specify real-valued signals over continuous time, including timing constraints. It is therefore a popular choice for embedded and cyber-physical systems. STL also include *robustness semantics*, which instead of giving a true/false evaluation of whether a signal satisfies a temporal logic formula, gives a quantitative number on how robustly the formula is satisfied. Hence, STL offers both a syntax to specify behaviours and a metric for evaluating the compliance to the behaviour, which has proven to be a powerful combination.

Another class of specification languages are *set-based*, building on mathematical set-theory. A modern and prominent example of this is Event-B (Abrial 2011), which combined with the free software tool Rodin supports all the steps in Figure 1. Event-B is appropriate for specifying both structure and discrete-event behaviours. The predecessor of Event-B, called B Method, was used to formally specify, verify, and automatically generate 86 000 lines of code for the driverless metro in Paris, resulting in a system for which no bugs have been found (Lecomte et al. 1991).

A final important class of FMs is called *theorem provers*. Here, specifications are written as mathematical statements and there exists tools to generate mathematical proofs of the statements. Theorem provers can be divided into automatic provers, such as Z3 (de Moura and Björner 2008), and interactive provers, such as Isabelle (Paulson 1994), where the user interacts with a computer tool to build a proof.

Design iterations

Having created a high-level specification, the next step in a formal development process is to refine the specification in a series of design iterations. Each design iteration adds more detail to the specification and brings it closer to something which can be realized in hardware and software. An important activity for each design iteration is the *refinement checking*, which aims to verify that a refined specification still contains all the properties of the higher-level specification. In other words, if a system meets the refined specification, it should also meet the higher-level specification. Many FM tools include automated refinement checking.

Another important step in the design iterations is *consistency checking*. This involves checking a specification for contradictions. The most obvious form of a contradictory specification is the statement “**(a) AND (NOT a)**”. In a complex specification however, contradictions can be subtle and hard to detect. A classic example is interface incompatibility. This can lead to expensive redesigns at a later stage in the development process. Many FM tools also support automated consistency checking.

Realization

The next step in a formal development process is to create an implementation in hardware or software based on the formal specification. In its most lightweight form, this can simply involve manual programming from the specification. Although this does not use the full potential of a formal development process, many find it easier to produce correct code when writing from a clear and unambiguous specification where fundamental design flaws have been removed at the design stages.

For some specification languages, there also exist tools which can automatically generate a *correct-by-construction* implementation from the specification. Examples of this include generation of computer code in various programming languages and synthesis of controllers which control a system to meet the specification.

Formal verification

The final step in the formal development process is the *verification*. Here, we define verification as checking that an implementation satisfies the lowest level specification. The most prominent formal verification technique is *model checking*. The input data to a model checking tool is the system to verify and a specification to verify against. The specification is usually in the form of a temporal logic formula. The model checker achieves an exhaustive verification by checking all possible executions of the system against the properties defined in the formal specification. The result is either a verification if no violating behaviour is found, or a falsification with a corresponding counter example if a violating behaviour is found. Some prominent model checkers are SPIN (Holzmann 1997), NuSMV (Cimatti et al. 2002) and UPPAAL (Larsen et al. 1997).

Automated theorem provers, as introduced earlier, can also be used to obtain a formal verification by for instance proving the correctness of an algorithm. Finally, *reachability analysis* (Asarin et al. 2007) is worth mentioning. This technique identifies the set of reachable states from any state in a system. By defining an unsafe set, this can be used to formally verify safety by showing that the unsafe set is not reachable from any state.

CURRENT PRACTICE IN THE MARITIME INDUSTRY AND NEEDS GOING TOWARDS AUTONOMY

Introducing novel and disruptive autonomous technology will alter the way ships are designed and operated. This has the potential to reduce some risks related to manned operations. At the same time, new risks emerge that need to be identified and mitigated to ensure safety. In this section the current practice for verification of maritime control systems is presented and some challenges and needs going towards autonomy are discussed.

To illustrate the current practice for verification of maritime control systems, we will use the certification process for dynamic positioning (DP) systems as an example. DP systems are complex automatic control systems involving both sensing, actuation, and computer systems (Sørensen 2011). The certification process for DP systems has been developed based on decades of experience. We therefore believe that DP systems represent a relevant example of state-of-the-art practice for verification of complex maritime control systems.

The international guidelines for DP systems are published by the International Maritime Organization (IMO) in IMO 1580 (IMO 2017). These guidelines provide the requirements for achieving a certification in form of a Dynamic Positioning Verification Acceptance Document (DPVAD). The guidelines specify both functional and operational documents in addition to requirements for verification activities. They are mostly concerned with the computer systems, power system, thrusters, and sensor systems. Classification societies publish class notations based on IMO 1580, which add more detailed and possibly additional requirements that strengthen safety and performance.

The overarching safety philosophy for IMO 1580 is heavily inspired by *functional safety*. This means that safety is promoted by identifying a set of *failure modes* and providing safety functions to ensure that no single failure can lead to a loss of position. The safety function is very often implemented by requiring redundancy, such as having two or three separate DP computers, three independent position reference sensors and two segregated power systems. The guidelines require that a *Failure Mode and Effect Analysis* (FMEA) should be carried out, which involves a systematic analysis of the DP system listing all failure modes and demonstrating the safe response. The survey and testing activities required by the guidelines involve a complete, physical survey of the DP system during commissioning and address primarily hardware components. This includes FMEA proving trials, where failure modes are simulated, and proper response by the DP system is verified. Furthermore, periodical testing at least every five years is required in addition to annual surveys. Proper testing and verification of software are not covered by FMEA testing alone. This may be characterized as a weakness with the existing mandatory regulations for software intensive systems such as DP (Johansen et al. 2007, Smogeli and Skogdalen 2011). Attempts at closing this gap includes both product assurance through e.g. Hardware-in-the-Loop testing (DNV 2013) and process assurance through SW development inspired standards such as ISDS (DNV 2017), but neither of these approaches have become mainstream in the assurance of maritime control systems.

Looking towards MASS, the need for new methodology for handling complex, software intensive systems is even more pressing. The challenges of the current, prescriptive classification regime are also acknowledged in the DNV class guideline for autonomous and remotely-operated ships (DNV 2018). A functional safety philosophy alone is not adequate when the level of autonomy increases. A key attribute of autonomous systems is that they interact with dynamic, unstructured, and uncertain environments. A consequence of this is that an autonomous vessel could exhibit unsafe behaviour without any equipment failure, for instance due to an unexpected environmental interaction or insufficient situational awareness. This challenge can be addressed by a complementary safety philosophy called *Safety Of The Intended Functionality* (SOTIF). Looking to the automotive industry, autonomous and advanced automation systems are now certified both for functional safety through ISO26262 (ISO 2011) and SOTIF through ISO21448 (ISO 2019), and it is likely that a similar development must take place for MASS.

Another consequence of interaction with dynamic, unstructured and uncertain environments is that the number of possible scenarios an autonomous system can encounter becomes enormous. Relying on testing alone, which even with massive scaling through parallelized simulators, only can analyse a limited number of scenarios, will therefore not be sufficient to ensure safety. Instead, there is a need for methodology to design systems with mathematically proven safety guarantees which reduce the span of possible scenarios.

Another gap in the current practice is the lack of focus on software failures. Autonomous vessels will be inherently software intensive, and their software will likely have significantly higher complexity than current systems. The current practice has a high focus on hardware failures, which have a different nature than software failures. The IEC 60050 standard defines a software failure as a manifestation of a dormant software fault, and a software fault is defined as a state of a software item that prevents it from performing as required. Software faults can for instance come in form of specification faults, design faults, programming faults, compiler-inserted faults, or faults introduced during software maintenance (IECTC 2002). A

software component may work perfectly for several years before a particular combination of input and internal state causes a software failure. Most of these types of failures are not possible to detect and remove by surveys and onboard FMEA testing only. Simulation-based testing through a Hardware-in-the-loop (HiL) setup has been practiced for DP systems for almost two decades as a voluntary additional service. In HiL testing, the DP software, running on the target hardware, is connected to a simulation model of the vessel (digital twin) and its environment through a HiL interface. This enables more targeted, detailed and extensive testing of the control software compared to onboard testing. It is possible to conduct a complete virtual sea trial before the targeted control system is installed on the ship. The experience from HiL testing of DP software has uncovered a large number of critical software bugs both in the core software and in configuration of particular DP vessels (Smogeli 2015). Large scale simulation-based testing is expected to be a key methodology for verification of MASS (Pedersen et al. 2020). This requires formalization and automation in the scenario generation and selection, simulation evaluation and coverage assessment (Torben et al. 2022). There is also a need for methodology to produce software which is correct-by-construction, which furthermore will result in reducing the test space.

Experience from DP vessels has also shown that system integration is difficult and error prone. Integration often receives little attention in the design stage, where changes are easy and cheap to make, and is instead delayed until commissioning and sea trials. This was the motivation for creating the Open Simulator Platform (OSP) which provides means for sharing simulation-models under a standardized interface while protecting vendor intellectual property (IP) (Smogeli et al. 2020). As the complexity and criticality increase further with MASS, there is clear a need for a structured and formalized integration processes.

Recent advances in Machine Learning (ML) are a key enabler for autonomy. The use of ML methods, which learn from data instead of being programmed from a specification, has enabled engineers to solve problems which are hard to specify completely. Computer vision is a classic example of this. However, ML based software also introduces major challenges for safety verification, as they have a strong black-box characteristic and lack of explainability. In addition, the input may be high-dimensional, such as all the pixels in an RGB image. This renders traditional input-output testing impractical to get sufficient test coverage. Clearly, there is a need for new methods for verification of ML-based software (DNV 2020).

Finally, we will mention the challenge due to lack of experience with autonomous vessels. Going autonomous represents a step change both in terms of the technology and the operations, which means that there is very limited experience, guidelines and best practices to build on. The automotive industry has approached this by releasing millions of cars driving autonomously under human supervision and continuously collecting data and building experience. As the number of MASS will be far less than the number of autonomous cars, combined with the tendency to use customized components and doing one-of-a-kind builds, we cannot expect to gain large-scale experience in this way. This necessitates increased efforts in the design and verification phases.

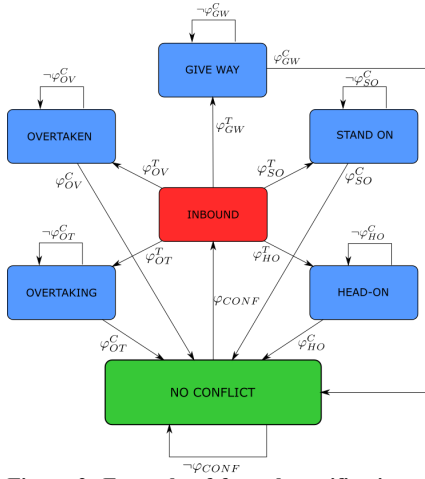
APPLICATIONS FOR FORMAL METHODS TO MASS DESIGN AND VERIFICATION

In this section we will demonstrate how FMs can be used to address some of the needs discussed in the previous section by presenting three specific use cases: Temporal logic specification of COLREG, contract-based design and automated simulation-based testing.

Temporal Logic Specification of COLREG

Safe interaction with other vessels is a challenging and critical aspect of MASS design and verification. Currently, this is regulated by the Convention on the International Regulations for Preventing Collisions at Sea – COLREG (IMO 1972), which specify a set of maritime traffic rules for collision avoidance. COLREG were designed for manned vessels and employ terms such as “due regard”, “appropriate distance” and “ample time”, and are therefore stated in a format that is not fit for computer parsing. To design collision avoidance algorithms and to enable automatic evaluation of COLREG compliance during verification, there is a need for machine readable COLREG. FMs stands out as a good candidate for formal specification of COLREG. An approach for this has been proposed in Torben et al. (2022) using STL, as shown in Figure 2. A similar approach, using the closely related Metric Temporal Logic (MTL) is proposed in Krasowski and Althoff (2021).

It may also be appropriate to develop a new collision avoidance protocol partially replacing COLREG, which is designed to be machine readable from the beginning. Using a formal language to specify the protocol also enables formal verification of safety properties. A similar development can be observed in the aviation industry, where the traditional TCAS protocol has been superseded by the ACAS X protocol, which has been formally specified and verified (Jeannin et al. 2017).



$$\begin{aligned} \varphi_{colreg} &= \square (HO \rightarrow \varphi_{HO} \wedge GW \rightarrow \varphi_{GW} \wedge OT \rightarrow \varphi_{OT}) \\ \varphi_{HO} &= t_{CPA} \leq t_{CPA,turn} \rightarrow \beta_r \in [-170^\circ, -10^\circ] \\ \varphi_{GW} &= t_{CPA} \leq t_{CPA,turn} \rightarrow d_{CPA} \geq d_{CPA,min} \\ \varphi_{OT} &= t_{CPA} \leq t_{CPA,turn} \rightarrow d_{CPA} \geq d_{CPA,min} \\ \varphi_{OT}^C &= \varphi_{OV}^C = \varphi_{GW}^C = \varphi_{SO}^C = \varphi_{HO}^C = t_{CPA} \leq 0 \\ \varphi_{NC}^C &= d_{CPA} \leq d_{CPA,min} \wedge t_{CPA} \leq t_{CPA,min} \end{aligned}$$

Figure 2: Example of formal specification of COLREG using STL. The left figure shows a finite-state machine for selecting the COLREG situation (HO = Head-on, GW = Give way, SO = Stand-on, OT = Overtaking, OV = Overtaken). Each situation has a trigger condition based on the sectors for the heading and bearing of an incoming vessel. To the right, a set of STL formulas are given for the COLREG situations which require explicit action by own ship. The reader is referred to Torben et al. (2022) for more details.

Contract-based design

Modularity in design is a key success factor for managing complexity. This is motivated by the “divide and conquer” problem solving strategy, where a complex problem is broken up into a set of simpler subproblems, and the solutions of the subproblems are combined to solve the complex problem. Moreover, having a good framework for system integration of modular entities is a great benefit when interfacing with third-party systems and when integrating Commercial Off-The-Shelf (COTS) components. A challenge is to ensure that unsafe interactions across modules are sufficiently accounted for and mitigated.

Contract-based design is a formal approach for building modular systems (Sangiovanni-Vincetelli et al. 2012). Each modular unit is called a *component*, having a clearly defined interface in terms of inputs and outputs. Each component has a formal *contract* in assume-guarantee form. Verifying a single component involves producing evidence that the component satisfies a set of guarantees on its outputs, given that a set of assumptions on its inputs are satisfied. Contracts can be formulated in a formal specification language. When integrating a set of components, this reduces to checking the compatibility of the contracts. This activity is often termed *compositional reasoning*. The contract-based approach also supports design iterations, where a high-level abstract design is refined into more concrete and detailed designs, as illustrated in Figure 1.

Figure 3 shows an example of an autonomy system high-level component structure. The autonomy system component (grey) is intended to be connected to a DP system (not illustrated) which feeds the autonomy system with navigation data and controls the MASS to follow the motion reference given by the autonomy system. In addition, the autonomy system has environmental interactions in form of traffic, obstacles and weather. Looking inside the grey box, we can see that the abstract autonomy system component description is refined by three components, each with their own contract. To verify that this is a correct implementation, refinement checking is necessary. An informal description of refinement checking in assume-guarantee contracts is “assume no more, guarantee no less”. This means that the combination of assumptions of the three subcomponents are less restrictive than the assumptions of the abstract autonomy system component, and that the combination of guarantees of the subcomponents are strong enough to enforce the guarantees of the abstract component. When integrating the three components, the compositional reasoning involves proving that the contracts of connected components are compatible. An example of a contract for the motion planning component, written in natural language could be:

Assuming that the tracks of other vessels are accurate to 10m and that the navigation data are accurate to 1m, the motion reference is guaranteed to always have a distance of at least 50m to other vessels.

The compositional reasoning would then require verifying that the contract of the situational awareness component actually guarantees an accuracy of less than 10m on its tracks, and that the contract for the DP system guarantees navigation data with accuracy less than 1m. The refinement checking would involve verifying that the distance to other vessels guaranteed by the motion planning component is less than the distance guaranteed by the abstract autonomy system component. This shows how refinement with refinement checking can project high-level requirements onto lower-level components in a coherent way. If contracts are written in a formal specification language, both the refinement checking, and the compositional reasoning can be performed by an automated theorem prover.

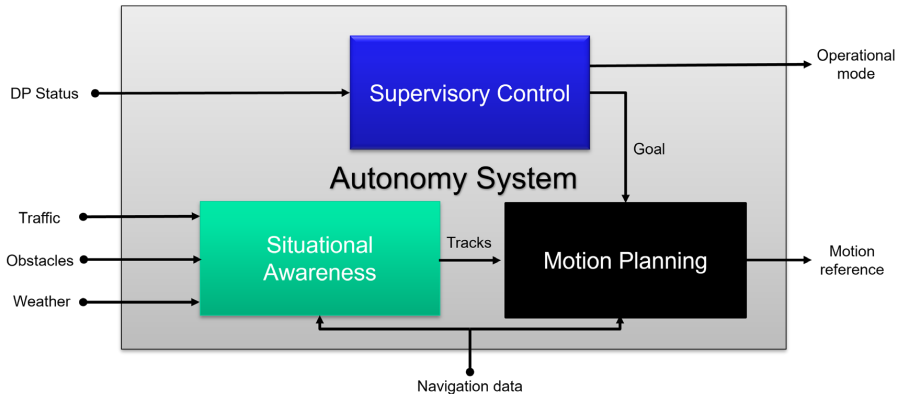


Figure 3: An example of a component structure for a high-level description of an autonomy system.

Automated simulation-based testing

In the final example we will show an example of how formal specification can be used to automate simulation-based testing. Evaluating the results of a simulation is often a non-trivial exercise which typically has been performed manually. If the testing scales to thousands of simulations, it is evident that manual evaluation is not practical, and for millions of simulations it is not practically possible. Torben et al. (2022) show how specifying requirements to test against in STL enables automatic evaluation of simulations using the STL robustness metric. The STL robustness metric gives a quantitative number on how robustly a simulation satisfies a requirement. If the robustness is greater than zero, the simulation satisfies the requirement, and if the robustness is less than zero, the simulation violates the requirement. The magnitude of the robustness is a measure of how much a signal can change without violating the requirement. Efficient algorithms exist for STL Monitors, which evaluate a signal against an STL formula (Fainekos et al. 2009).

The automatic testing methodology of Torben et al. (2022), shown in Figure 4, combines an STL monitor with a Gaussian Process (GP) model, which is used for smart scenario selection and coverage assessment. The user inputs an STL requirement and a test case. The test case is parametrized by a set of parameters which define the test space. The objective is to produce evidence that the system satisfies the STL formula to a desired probability level over the entire test space. The automatic testing algorithm selects a specific test case from the test space and runs this in the simulator. The STL monitor evaluates the simulation results against the STL requirement producing an STL robustness score. The GP model considers the STL robustness as an unknown function of the test case parameters and estimates the expected value and uncertainty of this function over the entire test space. The STL robustness score from a simulation is added to the GP model as an observation of this unknown function. This is shown to the right in Figure 4, where the GP estimate of the STL robustness is plotted in orange against the scenario parameters “*maneuver angle*” and “*obstacle speed*”. The black dots show observations. The GP model is used to select the next test case to simulate in a smart way, where it favours cases which have low robustness or high uncertainty. This adaptive sampling is prominent in Figure 4, where the black dots are denser in areas with low robustness. After each simulation, two termination criteria are checked: If a simulation which falsifies the requirement is found, the algorithm terminates in a falsified state. This is the case in Figure 4, where falsifying scenario is identified at speed = 10m/s and maneuver angle = 27 degrees. If no falsifying scenario is found, the termination is determined by the desired level of confidence. If, for instance, 99% confidence that the system satisfied the requirement is desired, then the algorithm terminates in a verified state if the lower 99% confidence interval of the GP model is greater than zero over the entire test space. This shows that the simulation-based testing is completely automatic after the test case and requirement are defined.

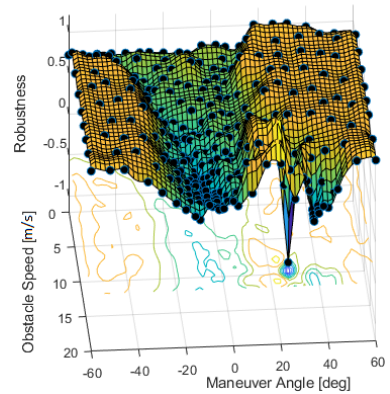
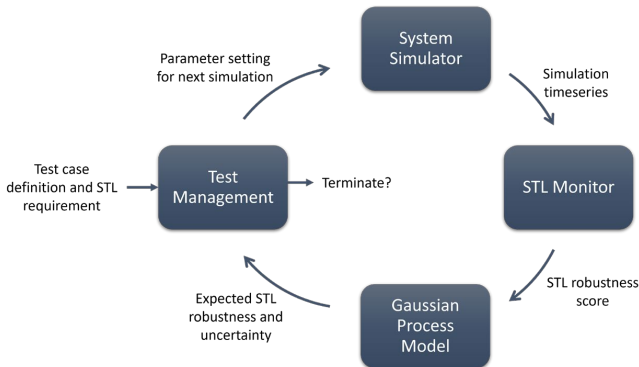


Figure 4: Left: Overview of the automatic test method of Torben et al. (2022). Right: Estimated STL robustness surface after a falsifying scenario is found.

LIMITATIONS OF FORMAL METHODS

The previous sections have shown some of the possibilities FMs have to offer. However, FMs not fit for all types of problems. In this section we will briefly discuss some limitations of FMs.

Scalability is a well-known limitation of FMs. Most FMs operate on discrete, finite-state models. As the number of variables in a model increases, the number of possible states increases exponentially. This is known as the *state space explosion* problem. Since a model checker exhaustively checks all possible executions, this limits the size of models which can be model checked. This has, however, been improved with recent innovations in model checking algorithms and the large increase in computational power.

FMs have traditionally also had limited support for systems with continuous dynamics, as they are often based on finite-state models. An approach to use FMs on continuous systems is to create a discrete approximation. However, this often ends up in huge state spaces which limits the size and accuracy of the discrete approximation. Several automated theorem provers have inherent support for continuous systems, by being able to use known theories on real-valued numbers in their reasoning.

Related to the two previous limitations is the *reality gap* problem. Formal verification is performed on a mathematical model of reality. In order for the verification of the mathematical model to have value as verification evidence, it needs to be an accurate representation of the real system. For some applications, such as logic circuits or software generated from a formal specification, the mathematical model is highly accurate. However, due to the scalability issues and limited support for continuous dynamics, some systems need to be simplified before they are subject to formal verification. Validating the fidelity of the simplified model is crucial in order to trust the results of a formal verification.

Finally, there is a long-standing challenge of limited uptake of FMs among professionals. We believe this is mainly caused by the perception that FMs are cumbersome, difficult and time-consuming. The majority of the background theory which FMs are built on originate from theoretical computer science and discrete mathematics, which is unfamiliar to many. This can make FMs seem intimidating to begin with. Also, taking on a formal development process often requires investing considerably more time in the design phase. However, this time is often returned in the verification and operational phases.

CONCLUSIONS

MASS are approaching reality, introducing a new level of complexity and criticality to maritime control systems. Given that the maritime industry already struggles with managing the complexity of current control systems, it is evident that new methods for development and verification are necessary to enable a safe and efficient introduction of autonomy. We have therefore proposed FMs as a candidate to aid the design and verification of safe MASS. We have discussed the needs the maritime industry faces and given three specific use cases demonstrating how FMs can be applied to meet these needs. Our conclusion is that FMs has the potential to meet many of the needs, despite some limitations. We therefore encourage further research into FMs for MASS.

ACKNOWLEDGEMENTS

The work is partly sponsored by the Research Council of Norway through the Centre of Excellence funding scheme, project number 223254, AMOS, and project ORCAS with project number 280655.

REFERENCES

- Abrial, Jean Raymond. *Modeling in Event-b: System and Software Engineering*. Cambridge University Press, 2011.
- Asarin, Eugene, et al. "Recent Progress in Continuous and Hybrid Reachability Analysis." *Proceedings of the 2006 IEEE Conference on Computer Aided Control Systems Design, CACSD, IEEE, 2007*, pp. 1582–87.
- Cimatti, Alessandro, et al. "NuSMV 2: An Opensource Tool for Symbolic Model Checking." *Lecture Notes in Computer Science*, vol. 2404, 2002, pp. 359–64.
- de Moura, Leonardo, and Nikolaj Bjørner. "Z3: An Efficient SMT Solver." *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer Berlin Heidelberg, 2008, pp. 337–40.
- DNV. *Rules for Classification of Ships Part 6 Chapter 22 - Enhanced System Verification*. 2013.
- DNV. *Recommended Practice: Integrated Software Dependent Systems (ISDS)*. 2017.
- DNV. *Class Guideline: Autonomous and Remotely Operated Vehicles*. 2018.
- DNV. *Recommended Practice DNVGL-RP-0510: Framework for Assurance of Data - Driven Algorithms and Models*. 2020.
- DNV. *RULES FOR CLASSIFICATION Ships Part 6 Additional Class Notations Chapter 3 Navigation, Maneuvering and Position Keeping*. 2021.
- Fainekos, Georgios E., and George J. Pappas. "Robustness of Temporal Logic Specifications for Continuous-Time Signals." *Theoretical Computer Science*, vol. 410, no. 42, Elsevier B.V., 2009, pp. 4262–91.
- Foster, Simon, et al. "Towards Deductive Verification of Control Algorithms for Autonomous Marine Vehicles." *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems*
- Holzmann, G. J. "The Model Checker SPIN." *IEEE Transactions on Software Engineering*, vol. 23, no. 5, 1997, pp. 279–95
- IECTC. *IEC 60050: International Electrotechnical Vocabulary*. 2002.
- IMO. *COLREGs - International Regulations for Preventing Collisions at Sea*. 1972.
- IMO. *Guidelines for Vessels and Units with Dynamic Positioning (DP) Systems. MSC.1/Circ.1580*, 2017.
- IMO. *Outcome of the Regulatory Scoping Exercise For The Use Of Maritime Autonomous Surface Ships (MASS)*. 2021.
- ISO. "Road Vehicles — Functional Safety." *ISO 26262*, 2011.
- ISO. "Road Vehicles - Safety of the Intended Functionality." *ISO 21448*, 2019.
- Jeannin, Jean-baptiste, et al. "Formal Verification of ACAS X, an Industrial Airborne Collision Avoidance System." *International Conference on Embedded Software*, 2015, pp. 127–36.
- Johansen, Tor A., et al. "Experiences from Hardware-in-the-Loop (HIL) Testing of Dynamic Positioning and Power Management Systems." *OSV Singapore*, 2007, pp. 41–50.

- Krasowski, Hanna, and Matthias Althoff. "Temporal Logic Formalization of Marine Traffic Rules." *IEEE Intelligent Vehicles Symposium (IV)*, 2021, pp. 186–92.
- Larsen, Kim G., et al. "UPPAAL in a Nutshell." *International Journal on Software Tools for Technology Transfer*, vol. 1, 1997, pp. 134–52.
- Lecomte, Thierry, et al. "Formal Methods in Safety Critical Systems." *Safety and Reliability*, vol. 11, no. 4, 1991, pp. 6–17.
- Maler, Oded, and Dejan Nickovic. "Monitoring Temporal Properties of Continuous Signals." *Lecture Notes in Computer Science*, vol. 3253, 2004, pp. 152–166.
- NMD. *Føringer i Forbindelse Med Bygging Eller Installerings Av Automatisert Funksjonalitet, Med Hensikt å Kunne Utføre Ubemannet Eller Delvis Ubemannet Drift*. 2020.
- Park, Jinwook, and Jinwhan Kim. "Autonomous Docking of an Unmanned Surface Vehicle Based on Reachability Analysis." *International Conference on Control, Automation and Systems*, 2020, pp. 962–66.
- Paulson, Lawrence C. *Isabelle: A Generic Theorem Prover*. Vol. 828, Springer Science & Business Media, 1994.
- Pedersen, Tom Arne, et al. "Towards Simulation-Based Verification of Autonomous Navigation Systems." *Safety Science*, vol. 129, no. December, Elsevier, 2020, p. 104799.
- Pnueli, Amir. "The Temporal Logic of Programs." *Annual IEEE Symposium on Foundations of Computer Science*, IEEE, 1977, pp. 46–57.
- Sangiovanni-Vincentelli, Alberto, et al. "Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems." *European Journal of Control*, vol. 18, no. 3, Elsevier, 2012, pp. 217–38.
- Shokri-Manninen, Fatima, et al. "Formal Verification of COLREG-Based Navigation of Maritime Autonomous Systems." *Lecture Notes in Computer Science, Software Engineering and Formal Methods*, 2020
- Smogeli, Øyvind. "Managing DP System Software - A Life-Cycle Perspective." *IFAC-PapersOnLine*, vol. 48, no. 16, 2015, pp. 324–34.
- Smogeli, Øyvind, et al. "Open Simulation Platform – An Open-Source Project for Maritime System Co-Simulation." *19th Conference on Computer and IT Applications in the Maritime Industries*, 2020, pp. 239–53.
- Smogeli, Øyvind, and Jon Espen Skogdalen. "Third Party HIL Testing of Safety Critical Control System Software on Ships and Rigs." *Offshore Technology Conference*, no. 2011, One Petro, 2011, pp. 839–45.
- Sørensen, Asgeir J. "A Survey of Dynamic Positioning Control Systems." *Annual Reviews in Control*, vol. 35, no. 1, 2011, pp. 123–36.
- Torben, Tobias R., et al. "Automatic Simulation-Based Testing of Autonomous Ships Using Gaussian Processes and Temporal Logic." *Journal of Risk and Reliability*, To be published, 2022.
- Yan, Rongjie, et al. "Formal Collision Avoidance Analysis for Rigorous Building of Autonomous Marine Vehicles." *Embedded Systems Technology*, edited by Yuanguo Bi et al., Springer Singapore, 2018, pp.

Paper E:

Evolution Of Safety In Marine Systems: From STPA To Automated Test Scenario Generation

Tom Arne Pedersen, Åse Neverlien, Jon Arne Glomsrud, Imran Ibrahim, Sigrid Marie Mo, Martin Rindarøy, Tobias Torben and Børge Rokseth

Journal of Physics: Conference Series
Volume 2311, No. 1, 012016

Evolution of Safety in Marine Systems: From System-Theoretic Process Analysis to Automated Test Scenario Generation

Tom Arne PEDERSEN^{1*}, Åse NEVERLIEN¹, Jon Arne GLOMSRUD¹, Imran IBRAHIM², Sigrid Marie MO³, Martin RINDARØY⁴, Tobias TORBEN⁵, Børge ROKSETH⁵

¹Det Norske Veritas (DNV), Group Research and Development, ²Det Norske Veritas (DNV), MI Advisory, ³Brunvoll, Department Engineer Marine Cybernetics, ⁴SINTEF Ocean, Ships and Ocean Structures, ⁵Norwegian University of Science and Technology (NTNU), Department of Marine Technology

Corresponding author: Tom Arne Pedersen; e-mail: tom.arne.pedersen@dnv.com; Tel.: +47 952 80 695

Abstract. The technological development ongoing in the maritime industry is making the ground for remotely and even autonomously operated vessels in the future. This is a result of increased data collection, processing and inter-connectivity capabilities. The industry is working towards increased safety, improved efficiency of the ship's operation, improved environmental performance and a more cost-effective shipping. New technologies are developed in order to reach these goals, and DNV as a Class society is developing frameworks for assurance of such systems. The certification of ships and vessels with a high degree of automation or autonomy needs an increased focus on software, an understanding of the human-to-machine interaction and the resulting ability to solve complex operations in a secure way. In this paper, a method for high-level risk analysis of the safety aspects of autonomous vessels combined with automatic simulation-based testing of a control system, is proposed.

1. Introduction

The Fourth IMO Greenhouse Gas study [1] estimated that the greenhouse gas (GHG) emissions from shipping has increased from 977 million tons in 2012 to 1,076 million tons in 2018. In 2018, IMO adopted an internal strategy [2] with a vision to reduce the GHG emissions from international shipping by at least 50% by 2050 and to reduce the carbon intensity of international shipping by at least 40% in 2030, towards 70% in 2050, compared to 2008. Even though there were an increasing demand for shipping from 2008 to 2014, the GHG emissions from international shipping decreased due to a period of rapid carbon intensity reduction [1]. The demand for shipping has continued to increase, but with a moderate improvement in carbon intensity, the emissions have declined since 2008, but in the period 2014-2018 the trend shows an emission growth, see Figure 1. To reach IMO's 2050 GHG reduction ambition, a large share of GHG reduction will have to come from alternative low-carbon fuels, energy-saving technology, and speed reductions of ships [1]. This is where autonomy can play a role as automated systems may gradually improve the environmental performance and enable more cost-effective shipping [3]. One way to lower the energy consumption of shipping is to reduce the energy consumption. For a ship, energy consumers can be divided into three groups: energy required for ship



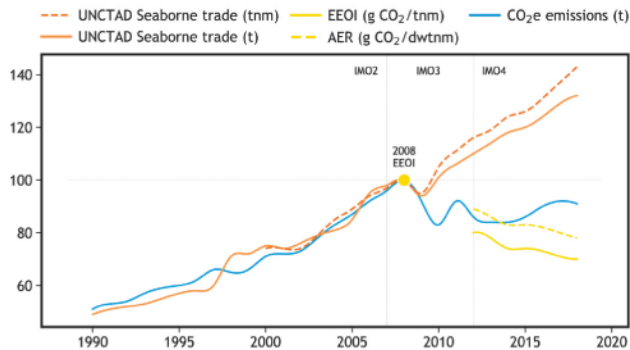


Figure 1: International shipping emissions and trade metrics, indexed in 2008, [1].

propulsion, hotel loads and for on-board operations [3]. In the first group, ship resistance is the main contributor and reduction of ship speed will change the carbon intensity. According to [3], two ships operation at 6 knots would in sum consume 30 to 50% less energy than one ship operating at 12 knots, depending on design and environmental conditions.

Ship navigation may be broken down in four sub-tasks: condition detection, external/internal situation awareness, action planning and action control. All ships must detect the actual situation, meaning own ship location, nearby objects that may pose a threat, status and conditions of on-board equipment that may potentially affect the vessel's ability to maneuver etc. The gathered information is then analyzed to give a complete situation awareness, before making a plan to best handle specific situations. Finally, the plan is effectuated. An autonomous navigation system (ANS) consists of several different controllers and algorithms, each controlling different aspects of the vessel. When putting these smaller control systems together, the resulting system is a complex system with emergent properties [4]. Emergent properties are properties that are not there initially, but may only be observed when the different controllers and subsystems are put together. In other words, they are not observable by looking at the individual pieces, but instead they emerge at a system level. In addition to being a complex system, the ANS also interact in a complex environment with an infinite number of different traffic situations.

In the perspectives of assurance and testing, ensuring that ANS algorithms are safe and do not cause accidents is crucial. Testing may be done in real life using the actual ship, in the virtual world using simulators, or in combination. Real-life testing only would be too time consuming, especially considering the of situations needed to be tested. In addition, critical situations may be too difficult or dangerous for real life testing, thus a combination of simulation-based and real-life testing would be the preferred solution. If enabled in the control system, simulation-based testing may also be performed by running the control system in closed loop with the simulator faster than real time, empowering accelerated testing.

[5] introduced System-Theoretic Process Analysis (STPA) as part of the verification and testing process of maritime automation systems such as dynamic positioning (DP) systems. STPA was used to identify safety constraints on system interactions and the constraints were translated into verification objectives.

[6] explored the feasibility of applying an STPA safety and security co-analysis on novel cyber-physical systems (CPS). They studied the STPA safety and security co-analysis to assess the risks of CPS collectively by adopting widely accepted security analysis methods into the STPA analysis process.

In [7], STPA was applied to an SAE (Society of Automotive Engineers) level four vehicle with the aim to uncover potential hazards that have not been discovered by traditional methods. The STPA

analysis resulted in 242 requirements, where 11 of these requirements were not handled in the current implementation.

A method for automatic simulation-based testing of control systems for autonomous vessels was proposed in [8]. The method formulated the requirements using formal Signal Temporal Logic (STL) and used a Gaussian Process (GP) model to estimate the robustness score and the uncertainty level for the untested cases in the simulation. The GP model was then used to guide the case selection towards cases with low robustness or high uncertainty. The methodology incrementally run new simulations until the parameter space covered to a desired confidence level, or until a case falsified the requirements.

In this paper, the results from [5], [6] and [8] are combined. A high-level risk analysis is performed using STPA. The outcome of the STPA is a set of loss scenarios with corresponding safety constraints for preventing losses. Signal Temporal Logic (STL) is used to express the STPA constraints as formal requirements.

2. Background Theory

2.1. System-Theoretic Process Analysis

Nancy Leveson introduced Systems-Theoretic Accident Model and Processes (STAMP) [9] by looking at safety as a control problem. She stated that safety can be controlled by applying sufficient constraints to a system. She also stated that reliable system components do not necessarily result in a safe system. In [10], Thomas argued that most major accidents today are not caused by component failure, but instead caused by complex and often unexpected interactions among components operating as designed. Flawed system design, human error or missing software requirements are also important factors, and to handle this, Leveson developed System-Theoretic Process Analysis (STPA) designed to analyze socio-technical systems characterized by complex human-machine interaction. Leveson argued that accidents would not occur if the system avoids hazards [11]. In the following, loss [11], accident and hazards [6] are defined:

Definition 1. Loss: A loss involves something of value to stakeholders. Losses may include a loss of human life or human injury, property damage, environmental pollution, loss of mission, loss of reputation, loss or leak of sensitive information, or any other loss that is unacceptable to the stakeholders.

Definition 2. Accident: An accident is an undesired or unplanned event that results in a loss.

Definition 3. Hazard: A hazard is a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident or a loss.

Losses can also relate to negated positive goals, e.g. an efficiency goal can be expressed as a loss of efficiency.

According to [11], it is prescribed to split the STPA analysis into 4 steps, as seen in Figure 2:

1. Define purpose of the analysis.
2. Model the control structure.
3. Identify unsafe control actions.
4. Identify loss scenarios.

The first step of the STPA analysis is to define what losses should be prevented. Not only safety goals, such as preventing loss of human life/injury, property damage and so on, but also efficiency requirements will come into play when performing the STPA analysis.

According to [6], it is not straightforward to transition into the second step of modelling the control structure. To mediate this challenge, they proposed to identify functional requirements as an intermediate step to convert the constraints into requirements to facilitate the development or assessment of the control structure. [6] states that STPA defines a control structure consisting of two important control system properties - i.e., controllability (control input and control actions) and observability (feedback and process model). Functional requirements include requirements for both

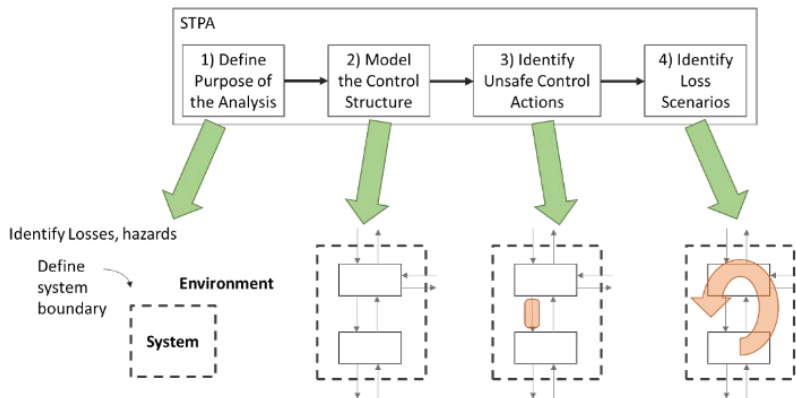


Figure 2: Overview of STPA analysis, adopted from [11].

properties. To control or mitigate the hazards, functional requirements address both the hazard and relevant control process.

The second step is to model the control structure of the system, given the system properties and system boundaries. The control structure consists of feedback control loops representing the functional relationships and interactions in the system. A set of interface arrows may be seen in lower part of Figure 2. The downwards pointing arrows generally contain control actions while the upwards pointing arrows generally contain feedback data. Controller responsibility may be assigned when the control structure is defined.

The third step examines how control actions may be inadequate and potentially lead to the hazards and losses defined in the first step. In STPA, these control actions are defined as unsafe control actions (UCA). In this paper, inadequate control action (ICA) is used to pinpoint that not only safety is the concern, and thus, in step 3, this is explored.

The fourth and final step identifies causal factors to how ICAs might emerge in the system. According to [11], loss or causal scenarios are identified to explain:

1. How incorrect feedback, inadequate requirements, design errors, component failures, and other factors could cause inadequate control actions and ultimately lead to losses.
2. How adequate control actions might be provided, but not followed or executed properly, leading to a loss.

By identifying the loss scenarios, additional requirements may be identified as well as mitigative actions that may additionally lead to design recommendations and changes. The scenarios may also be used for defining test cases and creating test plans to bring in evidence that the scenarios are managed properly by the control system.

2.2. Simulation-based testing

A simulator may be used to predict some aspect or behavior of a system or process to be used for different purposes, e.g. understanding a process, documentation or for collecting evidence demonstrating some goals or needs. Testing is an important contribution in gaining assurance and finally trust in cyber-physical systems. The following aspects of testing are relevant for simulation-based testing (inspired by [12] and [13]):

1. Testing of functional behavior
2. Testing of performance
3. Testing of robustness

2.2.1. Testing of functional behaviour Functional testing may be described as verifying the existence of the functions related to the planned operation and the needed functional interaction between the systems and sub-systems. It can be viewed as verifying the functional suitability of the system. To perform functional testing, a clear and concrete specification is needed for the intended functions and interactions with the operating environment or other parts of the system. To trust the testing it is important that the simulation adequately simulates the needed functions, operational scenarios and interactions.

2.2.2. Testing of performance According to [12], performance testing is defined as “type of testing conducted to evaluate the degree to which a test item accomplishes its designated function within given constraints of time and other resources”. Functional and performance testing are not totally separable, but they are rather distinctly different aspects. The performance aspect is not always relevant when considering a function, but often functions cannot fully be tested without building an opinion of the performance and vice versa, testing performance without including the functional behaviour is not possible.

2.2.3. Testing of robustness The third aspect of testing is testing of robustness. Robustness is defined as “the degree to which a component or system can function correctly in the presence of invalid inputs or stressful environmental conditions” [13]. Invalid inputs may come from defects in a component or system such as a sensor, preventing it from working as intended. Environmental disturbance could be caused by wind, current, waves and weather conditions, but also lighting conditions. It may also be more discrete and random events, e.g. happening in the operating environment, by the system operators. Robustness may be related to system inadequacies in general, that the system is not being capable of handling the operative environment, either due to the operative envelope exceeding the system’s capability or that the system’s capability is reduced. When testing robustness, exploratory testing where the tester actively controls the design of new tests based on already performed tests [13], will be important as it is difficult on beforehand to know where defects or other inadequacies in the system may hide.

When simulator-based testing is used for collecting evidence that a specific control system is performing as intended in different situations, the simulator is connected to the control system, e.g. using a classical Hardware-In-the-Loop (HIL) [14] or Software-In-the-Loop (SIL) set-up. In both set-ups, the control system should not be able to detect any difference between being connected to the real asset or the simulator. HIL and SIL are discussed next.

2.2.4. Hardware-In-the-Loop testing A control system interacts with its surroundings through input/output (I/O) communication channels. These inputs are from operator stations, other control systems, and sensors measuring parameters and dynamical states. The inputs are used, together with the internal models in the control system, to calculate control signals which are sent to the actuators. In a HIL set-up, the actual I/O’s are replaced with simulated I/O’s from a HIL simulator running in real-time. The surroundings of the control system, such as the dynamical systems, sensors and actuators, are imitated by the HIL simulator. The HIL simulator responds to control signals and provide consistent measurements, see Figure 3.

2.2.5. Software-In-the-Loop testing SIL testing allows for testing of the behavior of the control system without physically connecting the hardware to the simulator. The control software is either running on emulated hardware or as a separate process on the computer running the simulation. As the control software is not running on real-time hardware, it may be possible to control time, such that the control system may run in sync with the simulation possibly faster than real time. When using simulation models to generate trust, it is important that the simulation models can be trusted, being the next topic.

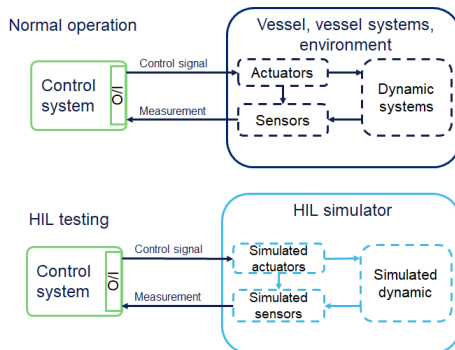


Figure 3: HIL testing conceptual setup, excluding other control systems and operators, courtesy of Marine Cybernetics AS.

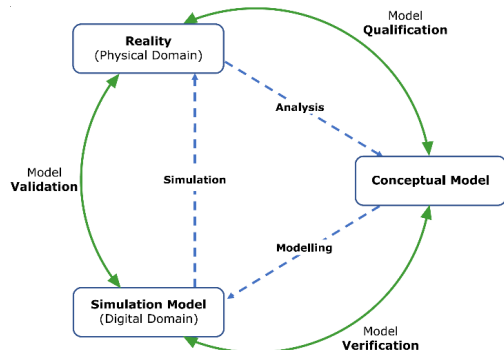


Figure 4: Relationship between reality, conceptual model and simulation model, together with verification and validation activities, [15].

2.2.6. Assurance of Simulation models In June 2021, DNV released a recommended practice (RP) providing a framework for assuring simulation models [15]. The RP defines what is necessary in terms of competence and assurance activities to minimize the risk of using a simulation model in a given use case to an acceptable level.

Before developing a simulation model, it is vital to first establish a conceptual model of the reality. The conceptual model gives a common understanding of the features to include, how to model these features and the high-level architecture of the model [15]. All three elements, the reality, the conceptual model and the simulation model, are important and essential to be included in the assurance of a simulation model, providing trust in the model and the predictions it makes about the reality in a specific use case (Figure 4).

More information about assurance of Simulation models is found in [15].

2.2.7. Automatic testing using Gaussian Processes and System Temporal Logic An automatic simulation-based testing approach was proposed in [8] which is briefly presented. For a deeper understanding of the proposed method, the reader is recommended reading the paper and the references therein.

An overview of the main components of the proposed testing methodology and how they interact can be seen in Figure 5. A simulator with all its simulator models, is described by a set of parameters, e.g. initial conditions, input signals and configurations. The results obtained from running a simulation is dependent on the value of these parameters. Let P be a parameter set defined by the Cartesian product $P = P_1 \times P_2 \times \dots \times P_n$. The parameter set contains all combinations for a simulator with n parameters and each parameter p_i has an associated P_i , where $i \in (1, 2, \dots, n)$. Testing all the simulations in P may not be relevant nor possible. Instead, a case Σ is defined as a collection of k parameters p_1, p_2, \dots, p_k with corresponding parameter sets P_1, P_2, \dots, P_k . Normally, one would keep most of the parameters in a simulation constant, and only let a few of the parameters vary. For each test case, the fixed parameters are described, as well as the parameters that may vary by the range values they may take. Each sub-case will then hold a unique set of varying and fixed parameters.

Figure 5 shows an overview of the main components of the proposed testing methodology. The test case describes the complete simulation in addition to the Signal Temporal Logic (STL) metric used for the evaluation. If applicable, also other evaluation criteria's will be given as part of the test case. Prior to simulation an initial set of sub-cases are generated using Latin Hypercube Sampling (LHS) [16] by splitting the predefined parameter space into n_{seed} slots, LHS selects samples such that no two samples occupy the same hypercube. The initial sub-cases are simulated, evaluated and the used to

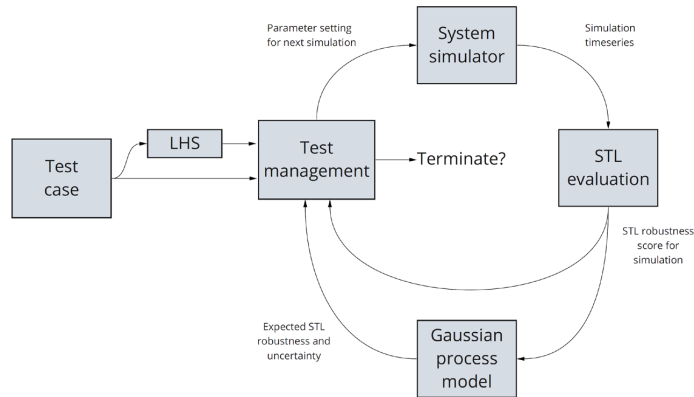


Figure 5: An overview of the main components of the proposed testing methodology and how they interact, adopted from [8].

establish an initial prior of the Gaussian process (GP) model. New parameter sampling points are then selected based on an updated prior of the GP model using a combination of

- points with low expected robustness, and
- points with large uncertainty.

By letting κ be an *exploration vs. exploitation* trade-off parameter, the next parameter set to simulate may be given by

$$\mathbf{p}_{next} = \operatorname{argmin}_{\mathbf{p} \in \mathcal{P}} \bar{\rho}(\mathbf{p}) - \kappa \sqrt{\operatorname{var}(\rho(\mathbf{p}))}, \quad (1)$$

where $\bar{\rho}(\mathbf{p})$ is the expected robustness and $\operatorname{var}(\rho(\mathbf{p}))$ is the associated covariance. The choice of κ may affect the number of simulations needed to obtain a sufficient confidence level. For small values of κ , the search is guided towards areas with low robustness which is good for fast falsification. With large values of κ , the search is guided towards unexplored areas with high uncertainty which is good for fast verification. Any test cases producing a low STL robustness metric indicates a falsification. If the expected STL robustness metric and uncertainty is above a certain limit, verification is concluded as a success. This will continue until any sub-case is falsified or all sub-cases are verified.

A brief description of STL and GP are given next.

2.2.8. Signal Temporal Logic STL is used to evaluate a real-valued signal [17] against a temporal logic formula to see if it satisfies a specific behavior by using predicates, is defined as:

$$\pi ::= f(y) \leq c, \quad (2)$$

where $f(y)$ is a scalar, real-valued function which maps the input signal y (an output from the simulation) to a real-valued scalar, and c is a real-valued scalar. The logical predicates can either be true or false.

To explain formulating STLs, the following example is given. Assuming the speed of a vessel is controlled by a speed controller, using speed feedback from three different sources. If one source fails, the controller shall still be able to control the vessel at a constant speed. This may be expressed using STL as

$$\varphi_{speed} = \square \left(|U_{vessel} - U_{ref}| \leq C_{\Delta} U_{vessel_{max}} \right), \quad (3)$$

where \square is the *always* operator such that φ is true if φ is true for all times $t \in T$, U_{vessel} is the vessel speed over ground and U_{ref} is the vessel speed reference.

One advantage with STL is the STL robustness metric which gives a quantitative indication of how robustly a signal satisfies an STL formula. Using predicates, STL provides a language to specify

behaviors of the system, and with the robustness metric, STL measures how well a behavior satisfies this formula.

A metric and signed distance are key concepts in STL defined next.

Definition 4. Metric [18]: A metric on a set S is a positive function $d: S \times S \rightarrow \mathbb{R}_{\geq 0}$ such that

$$\forall s, s' \in S, d(s, s') = 0 \Leftrightarrow s = s', \quad (4)$$

$$\forall s, s' \in S, d(s, s') = d(s', s), \quad (5)$$

$$\forall s, s', s'' \in S, d(s, s'') \leq d(s, s') + d(s', s''), \quad (6)$$

In this paper the Euclidean metric $d(s, s') = \|s - s'\|$ has been used.

Definition 5. Signed Distance [18]: With the metric d , the distance of a point $s \in S$ from a set $A \subseteq S$, the Signed Distance is defined as:

$$Dist_d(s, A) := \begin{cases} -\inf\{d(s, s') | s' \in A\} & \text{if } s \notin A \\ \inf\{d(s, s') | s' \notin A\} & \text{if } s \in A \end{cases} \quad (7)$$

The $Dist_d(s, A)$ is the shortest distance from s to any point in A . The signed distance states the robustness of $s \in A$, meaning how robustly a point belongs to a set. The signed distance is positive or negative in case a point is within or outside the set, respectively. The magnitude represents how far away s is from the boundary of A .

2.2.9. Gaussian Process model A Gaussian Process model is used to estimate the STL robustness metric the parameter combinations not already tested. According to [19], a GP is a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions defined by its mean function $m(x)$ and covariance function $k(x, x')$ as

$$f \sim \mathcal{GP}(m, k), \quad (8)$$

where f is a stochastic function, and $f(x)$ is the value of the function f at the location of the input argument x . There exists several covariance functions, also known as *kernel functions*, where the squared exponential covariance function probably is the most commonly used, given as

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma^2 \exp\left(-\frac{1}{2l^2} \|\mathbf{x}_i - \mathbf{x}_j\|^2\right). \quad (9)$$

It has two parameters, the variance σ^2 and the length scale l , which represents how much $\|x\|$ has to change to significantly change $f(x)$.

Assume an input $\mathbf{x}_i \in \mathbb{R}^{n_i}$, an output $y_i \in \mathbb{R}$ and a noisy observation set $D = \{(\mathbf{x}_i, y_i) | i = 1, \dots, n\}$, then consider the unknown function $y = f(\mathbf{x}_i)$ which can be learned from the observations (\mathbf{x}_i, y_i) . Assuming the noise in the observations is independent and Gaussian distributed, then

$$y(\mathbf{x}_i) = f(\mathbf{x}_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma_n^2), \quad (10)$$

where $\epsilon_i \in \mathbb{R}$ is a zero-mean white Gaussian noise with variance σ_n^2 .

With the defined observation set, the value of $f(\mathbf{x})$ at n_* test points are to be predicted, given by $\mathbf{f}_* \in \mathbb{R}^{n_*}$. A joint Gaussian probability distribution can then be defined, where \mathbf{y} is the known function values of the training set and \mathbf{f}_* is a set of function values corresponding to the test set inputs X_* :

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \sigma_n^2 & K(X, X_*) \\ K(X, X_*) & K(X_*, X_*) \end{bmatrix}\right), \quad (11)$$

where $K(X, X)$ is the $n \times n$ matrix training set covariances, $K(X, X_*)$ and $K(X_*, X)$ are the training-test set covariances and $K(X_*, X_*)$ is the test set covariance.

For prediction, a Bayesian inference approach is used to calculate the conditional probability distribution for \mathbf{f}_* given \mathbf{y} . This results in the distribution

$$\mathbf{f}_* | X, \mathbf{y}, X_* \sim \mathcal{N}(\bar{\mathbf{f}}_x, cov(\mathbf{f}_*)), \quad (12)$$

where

$$\bar{\mathbf{f}}_x = \mathbb{E}[\mathbf{f}_* | X, \mathbf{y}, X_*] = K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1} \mathbf{y} \quad (13)$$

and the covariance matrix is given as

$$\text{cov}(\mathbf{f}_*) = K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_n^2 I]^{-1}K(X, X_*). \quad (14)$$

The next step is to update the posterior predictions of the robustness values at each point in \mathbf{P} based on the observed robustness values. Given that n_{obs} observations have been made at points $\mathbf{P}_{obs} \in \mathbb{R}^{k \times n_{obs}}$, the observed robustness values are collected in the vector $\rho_{obs} \in \mathbb{R}^{n_{obs}}$. The robustness values at each point in \mathbf{P} and the observed points at \mathbf{P}_{obs} are jointly Gaussian with distribution

$$\begin{bmatrix} \rho_{obs} \\ \rho \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K_{obs} + \sigma_\epsilon^2 I & K_{cross} \\ K_{cross} & K \end{bmatrix}\right), \quad (15)$$

where $K_{obs} \in \mathbb{R}^{n_{obs} \times n_{obs}}$ is the covariance matrix for the observation points and $K_{cross} \in \mathbb{R}^{n \times n_{obs}}$ is the cross covariance between points \mathbf{P} and \mathbf{P}_{obs} . The conditional probability distribution of ρ given ρ_{obs} is given as

$$\rho | \rho_{obs} \sim \mathcal{N}(K_{cross}[K_{obs} + \sigma_\epsilon^2 I]^{-1}, K - K_{cross}[K_{obs} + \sigma_\epsilon^2 I]^{-1}K_{cross}^T), \quad (16)$$

where each point in \mathbf{P} have an associated expected value and uncertainty. In the remainder of this paper, let $\bar{\rho}(\mathbf{p})$ and $\text{var}(\rho(\mathbf{p}))$ refer to the expected value and variance of the STL robustness at point \mathbf{p} .

In [8], it is proposed to let the simulations run iteratively and update the GP until a desired confidence level is achieved, or until the requirements are falsified. For a sufficient confidence level, the criterion is that the minimum of the p_{conf} probability confidence interval of f_ρ is greater than zero. The search will be terminated in a *Verified* state when

$$p_{conf} = \min_{\mathbf{p} \in \mathcal{P}} \bar{\rho}(\mathbf{p}) - n_{conf} \sqrt{\text{var}(\rho(\mathbf{p}))} \geq 0 \quad (17)$$

where n_{conf} is defined as the number of standard deviations associated with the confidence level p_{conf} . Similarly, if the search process finds a case which falsifies the requirements, a case with robustness lower than zero, the search will terminate in a *Falsified* state.

3. Analysis

A line fishing vessel is used as a case for exploring the proposed method. A complete analysis of the control structure for the fishing vessel, is not the scope of this paper and instead a brief presentation is given.

The analysis starts with a description of the operation, then a summary of the results from the STPA analysis are presented. At the end, an example of how the result may be formulated as formal signal temporal logic requirement is given.

3.1. Operation

The long-line fishing vessel MS Geir is used as a use case in the SEAOPS project [20]. The vessel is today navigated by a captain, either manually or using a conventional autopilot. The autopilot is controlling the course of the vessel while the speed of the vessel is controlled using pitch and rpm levers. In the SEAOPS project, a controller is developed that shall automatically follow a path or control the course/speed for optimal setting of the long-line, and at the same time minimize wear and tear or damages to fishing equipment or vessel machinery and minimize fuel consumption.

The analysis focuses on the line setting phase. The vessels speed through water (STW) should be between 8.9 and 9.2 knots for optimal bait setting conditions. If the speed of the vessel is too high, some fishing-hooks might be set without bait., which can result in reduced catch. If the speed. is too low, fishing line setting time increases, resulting in lower efficiency

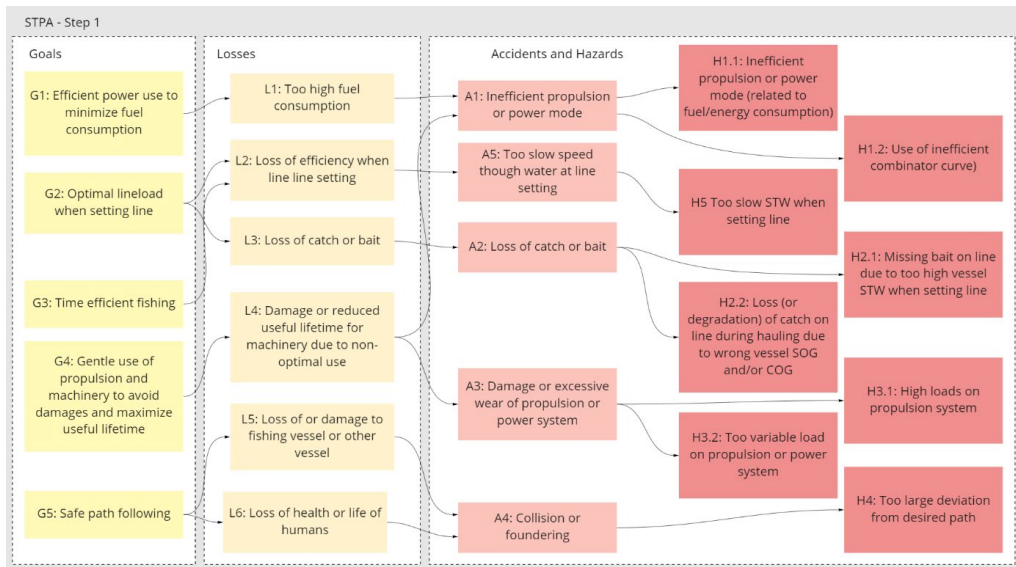


Figure 6: Goals - Losses - Accidents and Hazards

3.2. System Theoretic Process Analysis

3.2.1. *Step 1: Goals - Losses - Accidents and Hazards* During step 1 of the STPA, five operative goals were identified (see also Figure 6): G1 - Efficient power use to minimize fuel consumption, G2 - Optimal line load when setting line, G3 - Time efficient fishing, G4 - Gentle use of propulsion and machinery to avoid damages and maximize useful lifetime, and G5 - Safe path following. The corresponding losses were: L1 - Too high fuel consumption (G1), L2 - Loss of efficiency when line setting (G2). L3 - Loss of catch or bait (G2/G3), L4 - Damage or reduced useful lifetime for machinery due to non-optimal use (G4), L5 - Loss of or damages to fishing vessel or other vessels (G5) and L6 - Loss of health or life of humans (G5).

The detailed operation was analysed and accidents were identified and connected to the relevant losses. For each accident, hazards were identified:

- A1 Inefficient propulsion or power mode (L1, L4)
 - H1.1 Inefficient propulsion or power mode (related to fuel/energy consumption)
 - H1.2 Use of inefficient combinator curve
- A2 Loss of catch or bait (L3)
 - H2.1 Missing bait on line due to too high vessel STW when setting line
 - H2.2 Loss (or degradation) of catch on line during hauling due to wrong vessel SOG and/or COG
- A3 Damage or excessive wear of propulsion or power system (L4)
 - H3.1 High loads on propulsion system
 - H3.2 Too variable load on propulsion or power system
- A4 Collision or foundering (L5, L6)
 - H4 Too large deviation from desired path
- A5 Too slow STW line setting (L2)
 - H5 Too slow STW when setting line

In the following, H4: Too large deviation from desired path is selected presenting the analysis and for covering the most severe loss, see Figure 6.

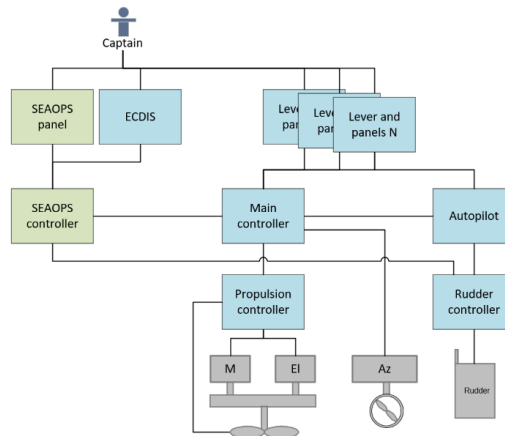


Figure 7: Overall control structure for MS Geir.

Treating the hazards as control problems, functional requirements for monitoring and control are identified:

- H4 Too large deviation from desired path
 - FR4.1 Desired path needs to be known.
 - FR4.2 Actual vessel position/heading/ course/speed need to be known.
 - FR4.3 Vessel position/heading/course/ speed need to be controlled.

The next step in the analysis is to identify the control structure.

3.2.2. Step 2: Control Structure The functional requirements identified in step 1 is mapped towards functions in the control system. The functional requirements connected to H4 are used as examples. FR4.1: Desired path needs to be known is mapped to be the responsibility of the captain, as he/she shall give way-point inputs to the SEAOPS controller ECDIS or similar. FR4.2: Actual vessel position/heading/course/speed need to be known may be estimated by e.g. a vessel state estimator in the SEAOPS controller based on feedback from the global positioning system (GPS). The last functional requirement is FR4.3 Vessel position/heading/course/speed need to be controlled, which is the control objective of the SEAOPS controller.

When all identified functional requirements are mapped as responsibilities of specific functions in the control system, the control structure can be developed by identifying the controllers, their control inputs, control actions and feedback information on which they depend on. The overall control structure for the fishing vessel MS Geir is shown in Figure 7 with a short description. The green boxes, the SEAOPS panel and the SEAOPS controller, will be developed during the project, and is target for the analysis and later testing. The blue boxes are existing panels, levers and control systems on board MS Geir, while the grey boxes are the components that are controlled by the on board control systems, e.g. main motor (M), electric motor (EI), azimuth (Az) and rudder. The captain controls the SEAOPS controller using the SEAOPS panel, while route plans are sent to the SEAOPS controller using ECDIS. The SEAOPS controller is connected to the main controller and the rudder controller. Other levers and panels and Autopilot are disengaged when SEAOPS controller is in control. The captain will always have the possibility to disable the SEAOPS controller and take direct control of the vessel. The SEAOPS controller will be responsible for deciding the optimal propulsion mode together with optimal combinator curve for (pitch/rpm) and controlling course and speed using the main propeller, rudder and azimuth.

3.2.3. Step 3/4: Inadequate Control Actions and Design-specific causes Inadequate control actions (ICAs) are identified in step 3. An ICA is defined as a control action that in a particular situation and

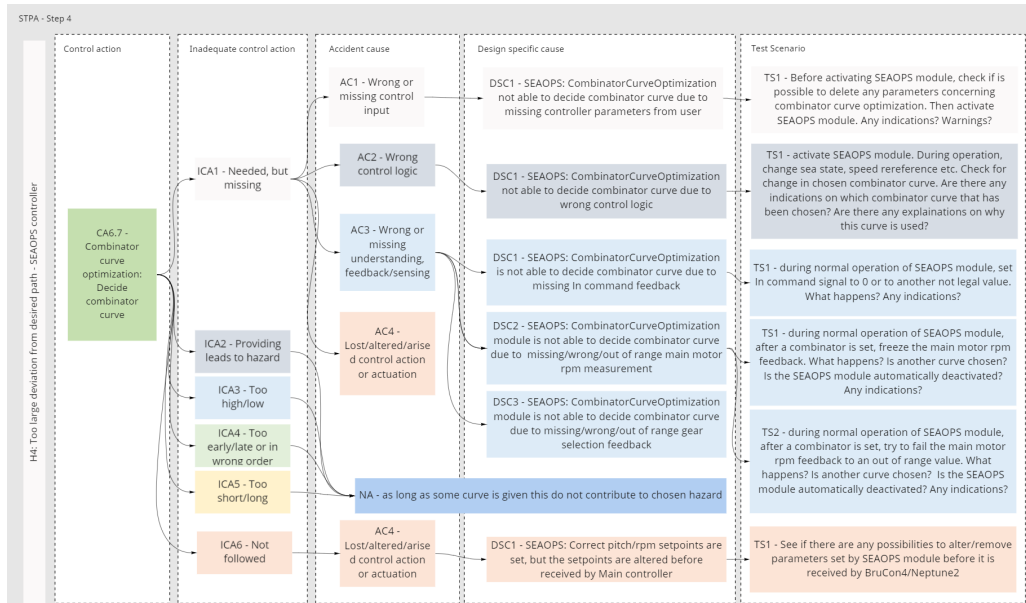


Figure 8: Step 4 - Design-Specific cause

in worst-case scenario leads to a hazard. [11] lists four ways a control action may be inadequate or unsafe:

- Not providing the control action leads to a hazards Providing the control action leads to a hazard.
- Providing a potentially adequate control action too early, late or in a wrong order leads to a hazard.
- The control action lasts too long or is stopped too soon leads to a hazard.

To further help the analysis, the following inadequate control actions were additionally added:

- Providing too high or too low control action leads to a hazard.
- Not following the control action leads to a hazard.

The last ICA may seem similar to the first ICA but covers the case if a proper control action is implemented but for some reason not followed. This is done for consistency in the analysis. If the control action is handled by a lower-level controller included in the analysis, the ICA can cover potential corruption of the information over the communication, but this overlaps with the accident cause 1 (AC1) explained later in this paper. If the lower-level controller is not included, the ICA can cover any case of inadequate implementation of a control action (see specifically accident cause 4 (AC4) also explained later). With identified ICAs, the controller constraints may be derived directly, which is the last step (4) in the analysis, that is to define causal scenarios or loss scenario which describes the causal factors that can lead to an inadequate control action and to hazards. According to [11], two types of loss scenarios should be considered:

- How would inadequate control action occur?
- How would control actions be improperly executed or not executed, leading to hazards?

The SEAOPS controller includes several control functions and the focus in the following is the control function *Combinator curve optimization: Decide combinator curve*. Parts of the analysis is shown in Figure 8. The list of ICAs are used together with the following defined accidents causes based on [11]:

- AC1 - Wrong or missing control input.
- AC2 - Wrong control logic.

- AC3 - Wrong or missing understanding/ feedback/sensing
- AC4 - Lost/altered/arised control action or actuation.

A design specific cause (DSC) answering ICA1 - Needed, but missing and AC1 - wrong or missing control input, may be written as

DSC1 - SEAOPS: CombinatorCurve-Optimization not able to decide combinator curve due to missing controller parameters from user.

The DSC may be tested using the following Test Scenario:

TS1 - Before activating SEAOPS module, check if is possible to delete any parameters concerning combinator curve optimization. Then activate SEAOPS module. Any indications? Warnings?

By combining ICAs with accident causes, it is possible to identify a set of DSCs which again may be used to define Test Scenarios. In this example, ICA2 to ICA5 have been defined as Not Applicable (NA), using the argument that if any curve has been chosen, then this will not lead to the chosen hazard which is H4 - Too large deviation from desired path. If the hazard had been something else, e.g. low efficiency, then ICA2 to ICA5 would have been used.

The identified Test Scenarios may then be used for testing the control system. Some of the test may be performed manually on the actual vessel, while others may form the basis for simulator- based testing. Some of the scenarios will also be suitable for automatic testing using the STL robustness metric, which is the next topic.

3.2.4. Development of STL robustness metric The hazard H4 - Too large deviation from desired path may be chosen for exploring the use of STL robustness metric together with the use of GP to guide the test sub-case selection. Looking into CA6.7, ICA1 and AC3, see Figure 8, the following DSC can be transformed to STL robustness metric:

DSC2 - SEAOPS: CombinatorCurve-Optimization module is not able to decide combinator curve due to missing/wrong/out of range main motor rpm measurement.

The steps needed for transforming the test scenario to test requirement and then to STL formulation are given in Table 1.

4. Conclusion

In this paper, the results from [5], [6] and [8] have been combined to develop evaluation requirements in addition to develop a method for automating exploratory testing to be used for assurance of complex system.

The next step will be to implement these requirements as evaluation criteria into a test system to demonstrate a procedure for simulation-based testing of the SEAOPS controller using a test setup with the controller connected in closed loop with a simulator. The use of GP to guide the selection of new sub-cases to test together with STL for evaluation is believed by the authors to be an efficient way to automate testing of complex systems such as autonomous navigation.

Acknowledgement

This work was supported by the Research Council of Norway through the project "SEAOPS", project number 10203745.

Table 1: From STPA to STL, CA6.7ICA1.AC3.DSC2.TSI

Test scenario	Set desired speed reference for line setting and activate SEAOPS module. After a combinator curve is set, freeze the main motor rpm feedback. What happens? Is another curve chosen? Is the SEAOPS module automatically deactivated? Any indications?
Test requirement	For all tests, the SEAOPS module shall be activated. Set a speed reference within line setting range. Freeze main motor rpm feedback. Check several different speed references, all within line setting range. For all rpm speed feedback, freeze the feedback's one by one. <ul style="list-style-type: none"> - Vessel speed should remain approximately constant after the failure has been initiated. - Propeller pitch should not be changed after the failure has been initiated. - Propeller rpm should not be changed after the failure has been initiated. - Check for warnings.
STL formulation	Parameterization: $U_{vessel,ref} \in [8.0, 9.5]$ [knots], where $U_{vessel,ref}$ is desired speed reference for the vessel. $C_{\Delta U_{vessel,max}}$ is maximum allowed speed variations, $C_{\Delta P_{max}}$ and $C_{\Delta \omega_{max}}$ are maximum allowed pitch and propeller speed variation, respectively. Parameter to be adjusted: $U_{vessel,ref}$ $\varphi_{speed} = \square(U_{vessel} - U_0 \leq C_{\Delta U_{vessel,max}})$ $\varphi_P = \square(P - P_0 \leq C_{\Delta P_{max}})$ $\varphi_{\omega} = \square(\omega - \omega_0 \leq C_{\Delta \omega_{max}})$ <p>U_{vessel}, P and ω are vessel speed, propeller pitch and propeller speed feedback, respectively, while U_0, P_0 and ω_0 are vessel speed, propeller pitch and propeller speed at the moment when the main motor rpm feedback to the controller was frozen.</p>

References

- [1] IMO, "Fourth IMO Greenhouse Gas Study," *International Maritime Organization*, 2020.
- [2] IMO, "The Initial IMO Strategy on Reduction of GHG Emissions from Ships," *International Maritime Organization*, 2018.
- [3] B. J. Vartdal, R. Skjong and A. L. St. Clair, "Remote-Controlled and Autonomous Ships," DNV, 2018.
- [4] O. I. Haugen, "Properties of evidence and evidence generation," *Marine Technology Society, Dynamic Positioning Conference*, 2018.
- [5] B. Rokseth, I. B. Utne and J. E. Vinnem, "Deriving Verification Objectives and Scenarios for Maritime Systems Using the Systems-Theoretic Process Analysis," *Reliability Engineering and System Safety*, 2018.
- [6] J. A. Glomsrud and J. Xie, "A Structured STPA Safety and Security Co-analysis Framework for Autonomous Ships," in *Proceedings of the 29th European Safety and Reliability Conference*, 2019.
- [7] G. Koelln, M. Klicker, T. Schmid and S. Schmidt, "System Theoretic Process Analysis for a Vehicle SAE Level Four," in *Proceedings of the 30th European Safety and Reliability Conference*, 2020.

- [8] T. Torben, J. A. Glomsrud, T. A. Pedersen, I. Utne and A. Sørensen, "Automatic Simulation-based Testing of Autonomous Ships using Gaussian Processes and Temporal Logic," *Journal of Risk and Reliability*, 2022.
- [9] N. Leveson, *Engineering a Safer World: Systems Thinking Applied to Safety*, The MIT Press, 2012.
- [10] J. Thomas, *Extending and Automating a Systems-Theoretic Hazard Analysis for Requirements Generation and Analysis*, PhD thesis, Massachusetts Institute of, 2013.
- [11] N. Leveson and J. Thomas, *STPA Handbook*, The MIT Press, 2018.
- [12] ISO, "ISO/IEC/IEEE 29119-1:2013 Software and systems engineering — Software testing — Part 1: Concepts and definitions," *ISO/IEC/IEEE*, 2013.
- [13] ISTQB, "Standard glossary of terms used in Software Testing," *International Software Testing Qualifications Board*, 2010.
- [14] R. Sjetne and O. Egeland, "Hardware-in-the-loop testing of marine control systems," in *46th Conference on Simulation and Modeling (SIMS 2005)*, 2005.
- [15] DNV, "Recommended Practice: Assurance of Simulation Models," *DNV-RP-0513*, 2021.
- [16] M. D. McKay, R. Beckman and W. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, pp. 239-245, 1979.
- [17] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," *In Lecture Notes in Computer Science*, p. 152–166, 2004.
- [18] G. Fainekos and G. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theoretical Computer Science*, pp. 4262-4291, 2009.
- [19] C. E. Rasmussen and K. I. Williams, *Gaussian Processes in Machine Learning*, the MIT Press, 2006.
- [20] "SEAOPS," January 2022. [Online]. Available: <https://www.dnv.com/research/review-2020/featured-projects/seaops-ship-operations-simulation.html>.
- [21] H. Rashmi, S. Yako, K. Post and S. Nuesch, "Systems Theoretic Process Analysis for Layers of System Safety," *INCOSE International Symposium*, vol. 29, pp. 895-909, 2019.

Paper F:

Development and testing of a risk-based control
system for autonomous ships

**Thomas Johansen, Simon Blindheim, Tobias Rye Torben, Ingrid
Bouwer Utne, Tor Arne Johansen and Asgeir Johan Sørensen**

Accepted for publication in Reliability Engineering and System Safety

Development and testing of a risk-based control system for autonomous ships

Thomas Johansen^{a,b,*}, Simon Blindheim^{a,c}, Tobias Rye Torben^{a,b}, Ingrid Bouwer Utne^{a,b}, Tor Arne Johansen^{a,c} and Asgeir J. Sørensen^{a,b}

^aCentre for Autonomous Marine Operations and Systems (NTNU AMOS), NTNU, Norway

^bDepartment of Marine Technology, Norwegian University of Science and Technology (NTNU), Trondheim, 7491, Norway

^cDepartment of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, 7491, Norway

ARTICLE INFO

Keywords:

Autonomous Systems
Risk Modelling
Ship Control Systems
Systems Theoretic Process Analysis (STPA)
Bayesian Belief Networks
Verification

ABSTRACT


This paper presents a method for designing and verifying a control system with risk-based decision-making capabilities to improve its intelligence and enhance the safe operation of autonomous systems. The decision-making capabilities are improved, compared to existing control systems, using a Bayesian Belief Network (BBN) that is derived from the systems theoretic process analysis (STPA) as a foundation for an online risk model, which represents the operational risk for an autonomous ship. Combined with an electronic navigational chart (ENC) module to get accurate information about the environment, this enables the ship to operate in a safe and efficient manner. In addition, the control system is verified against safety and performance requirements using a formal verification method, based on temporal logic and Gaussian processes. The proposed methodology is tested in a case study where the system's behavior is compared with an existing conventional (manned) ship on experimental data from two routes along the coast. The case study shows that the performance of the SRC with respect to the autonomous ship speed and maneuvering is similar to how the existing ship is operated. This means that the proposed methodology shows promising results with respect to developing autonomous ships with control systems and leads to intelligent and safe behavior.

1. Introduction

Although conventional ships have control systems for navigation, maneuvering, and power management, they are designed to rely on human input and supervision onboard. For example, Dynamic Positioning (DP) systems are used to maintain a ship's position or to maneuver the ship at low speeds with good accuracy. Nevertheless, a human operator must specify the mission and be ready to take over control if the automatic system fails. Power management systems (PMS) also have a high degree of automation to control electric power generation, power distribution, and prevent blackouts on ships.

There is currently no automation system that monitors or controls the complete ship's operation, replacing the crew onboard. For example, engine control systems may monitor the engine and shut it down if there is a failure, even if this compromises the safety and integrity of the ship. An example is the Viking Sky incident, where the diesel generators were automatically shutdown due to low lubrication oil levels in a severe sea state, which led to a complete blackout and nearly caused the cruise ship with almost 1400 people onboard to ground in storm conditions (NSIA, 2019). In general, for a ship to operate safely and autonomously, its control systems must be able to assess risk (currently the task of the crew onboard conventional ships). Hence, Utne et al. (2020) propose a control system framework that can assess and manage risk, replacing some of the cognitive judgments that the crew would normally make while sailing to improve the autonomous ship's decision making. Thieme et al. (2021) describe how risk analysis methods can be integrated with control systems and identify four areas for implementing this. Another approach is further demonstrated in Johansen and Utne (2022). A risk model represented by a Bayesian Belief Network (BBN), which is based on a systems theoretic process analysis (STPA), assesses navigational risks for an autonomous cargo ship while sailing as part of a supervisory risk controller (SRC) for high-level control of the ship. This risk model provides information that can be used as a basis for selecting the control mode, machinery mode, and setting control objectives while sailing. Bremnes et al. (2020,

*Corresponding author

 t.joha@ntnu.no (T. Johansen)

ORCID(s):

Abbreviations

AIS	Automatic Identification System	LNG	Liquefied Natural Gas
AMMS	Autonomous Machinery Management System	Mech	Mechanical
ANS	Autonomous Navigation System	MPC	Model Predictive Control
AP	Autopilot	MSO	Machinery System Operating
API	Application Programming Interface	PMS	Power Management System
AUV	Autonomous Underwater Vehicle	PTI	Power Take In
BBN	Bayesian Belief Network	PTO	Power Take Out
CONOPS	Concept of Operations	RIF	Risk Influencing Factor
CPT	Conditional Probability Table	ROC	Remote Operation Center
DP	Dynamic Positioning	SLAM	Simultaneous Localization and Mapping
ENC	Electronic Navigational Chart	SO	Ship Operating
FMEA	Failure Mode and Effects Analysis	SRC	Supervisory Risk Controller
GNSS	Global Navigational Satellite System	STL	Signal Temporal Logic
GP	Gaussian Process	STPA	System Theoretic Process Analysis
H-RIF	High-level Risk Influencing Factor	UCA	Unsafe Control Action
HiL	Hardware-in-the-Loop	USD	United States Dollar
HSG	Hybrid Shaft Generator	VHF	Very High Frequency
I-RIF	Input Risk Influencing Factor		

2019) presented a similar control system for autonomous underwater vehicles (AUVs) for under ice operations. In this case, the SRC was used to set the altitude set-point, velocity set-point, and control strategy such that the AUV could avoid collision while performing under-ice mapping with sufficient accuracy.

Relevant risk factors have also been discussed in Fan et al. (2020). A framework to identify navigational risk factors for autonomous ships is presented, but without any further application. Chang et al. (2021) combine Failure Mode and Effects Analysis (FMEA) with evidential reasoning and Bayesian Networks to quantify the risk level of major hazards related to autonomous ships. Johansen and Utne (2020) propose to use STPA to identify potential hazards for autonomous ships and discuss some methods for finding additional quantitative data to use in a risk model, but without building and using the model. STPA is also used in Valdez Banda et al. (2019) for hazard analysis on autonomous passenger ferries. This paper suggests safety controls to mitigate the identified hazards when designing the ship. Wróbel et al. (2018) use STPA to develop a model to analyze safety and make design recommendations for autonomous vessels. Chaal et al. (2020) propose a framework to model the ship control structure, based on STPA that can be useful to describe the functionality of the system.

Risk models have also been used to predict the loss of AUVs during missions (Brito and Griffiths, 2016; Loh et al., 2020a,b) and to manage uncertainty in these missions (Brito, 2016). However, none of these models are connected or implemented as part of the control system. Other papers have discussed risk as part of collision avoidance but use risk in a very general term and lack a direct link to risk analysis and risk modeling (Hu et al., 2017; Wang et al., 2019; Woo and Kim, 2020; Lyu and Yin, 2019; Li et al., 2021; Gil, 2021). Combining some selected risk aspects with Model Predictive Control (MPC) has also been proposed for collision avoidance systems (Tengesdal et al., 2020a,b) and emergency management but the risk metrics that are used in these studies are not based on risk assessment and are simplified so that they can be used in an MPC application (Blindheim et al., 2020).

A quantitative risk model can provide good and useful information for an autonomous control system if it includes reliable information about the ship's position and its surroundings. One option is to use tools such as Simultaneous Localization and Mapping (SLAM) that can be used for AUVs (Yin et al., 2021; Willners et al., 2021; Sandøy et al., 2020) operating in areas where localization and mapping are challenging. Mapping the environment is unnecessary for autonomous ships because position data are available from global navigational satellite systems (GNSSs), such as position and speed measurements, and electronic navigational charts (ENC) are available. GNSS measurements are already used in control systems, such as in DP controllers to provide position and speed measurements. ENC data have been used in decision making systems, such as path planners, for ship navigation (Mağa and Magaj, 2012). The

data can then be used directly in the planner, with limitations on extracting and presenting the data. To address these limitations, Blindheim and Johansen (2022) developed an open-source application programming interface (API) to process and display the data with high accuracy and in short computation time. Their paper shows how the API can be used for certain tasks, such as path planning based on a dynamic risk optimization. A simple risk metric based on wind speed and direction, and the distance to land is used when planning the route.

Developing better control systems is an important step towards realizing autonomous ships, which in turn is expected to improve safety at sea (Wróbel et al., 2017; de Vos et al., 2021). However, it is important to demonstrate that these ships are safe in operation to achieve approval from the authorities and public acceptance. This means that autonomous ships need to be tested in various scenarios and environmental conditions. Today, verification, validation, and certification in the maritime industry depend on type of ship and operation. On advanced offshore installations and ships, the ship and control system are thoroughly tested through simulations, scale testing, sea-trials, and Hardware-in-the-Loop (HiL) testing. Extensive and thorough tests are necessary to get the systems approved by class societies and coastal states (IMO, 2017). Suppliers usually test individual components on less advanced ships during commissioning and sea-trials.

The shift towards autonomous ships presents several challenges with respect to verification and testing. Both the complexity and criticality of the software systems increase. In addition, the control system interacts with a highly dynamic and unstructured operative environment, which causes the span of possible scenarios to become enormous. Autonomous systems typically use machine-learning software to some extent, which introduces its own set of challenges (see Torben et al. (2022b)). Therefore, there is a need for new methodology to formalize and scale the verification and testing efforts to new levels.

Several recent works have aimed to address these challenges. For example, Pedersen et al. (2020) propose a test system for autonomous navigation systems (ANSs) and show how it can be used to verify the performance of a collision avoidance system. Torben et al. (2022a) present an Autonomous Simulation-based testing framework and show how it can be used to verify a collision avoidance system. Xiao et al. (2021) propose a quantitative evaluation method to evaluate obstacle avoidance methods for unmanned ships. These studies indicate that although the test systems work, they only work through testing a very limited part of the control system. They also lack a description of how the testing should be integrated into the design process for autonomous ship control systems.

To summarize the gaps identified in the current literature, it is necessary integrate risk with control systems intended for autonomous ships to improve its high level decision making. In addition, these control systems need access to data from ENCs, and they need to be verified in a formal and systematic manner to ensure the necessary safety and performance. Hence, the overall objective of this paper is to present a novel and interdisciplinary methodology to develop an SRC for high level control of autonomous ships that bridges risk modeling, optimization, ENC, and formalized verification to achieve safer and more intelligent performance of autonomous ships.

The proposed methodology is tested and compared to an existing conventional-manned ship for different coastal routes to assess how the SRC handles failures in the ship's machinery and propulsion system. The main scientific contribution is the demonstration of how the intelligence of an autonomous control system can be improved by combining thorough risk analysis and modeling, detailed data from navigational charts, and novel verification methodology. Compared to existing control systems, this new approach makes it possible to handle a wider range of operations and situations, which reduces the need for human intervention and supervision. Even though the application in this paper is focused on autonomous surface ships, it is expected that the methodology will have relevance for other autonomous applications.

The rest of this paper is organized as follows. Section 2 presents the methodology for building and setting up the controller. Section 3 describes the case study. Section 4.1 and section 4.2 present the results from the case study. Sections 4.3-4.7 discuss how risk can be included in control systems, how to use ENC data, how to test the system, and it also describes some uncertainties in the controller and risk model. Section 5 concludes this paper and outlines further work towards highly autonomous ships.

2. Method

The SRC controller is developed through a five-step process, as shown in Figure 1. The SRC enables the controller to make risk informed decisions that emphasize both safety and efficiency when operating the ship. These decisions can (for example) determine the ship's operating machinery mode, control mode, or the speed reference for the proposed control system.

The ship and the operation are first described in detail and analyzed using an extended STPA to identify hazardous events that need to be included in the risk model. Thus, the STPA results are used as the basis for building the online risk model in step 2, which is represented here in terms of a BBN. The justification for using STPA combined with BBN is presented in Utne et al. (2020). For situation awareness, the risk model uses data from the ship's sensors and the control system to assess the current conditions. The ENC module is used to extract data from navigational charts with information about the area surrounding the ship. The ENC model is set up in step 3 based on the design requirements to provide the necessary data to the risk model and SRC. The SRC is then developed in step 4 based on the requirements identified in the system analysis and the STPA (step 1), and using data from both the risk model and ENC. Finally, the controller is verified against the performance requirements using the automatic simulation-based testing methodology.

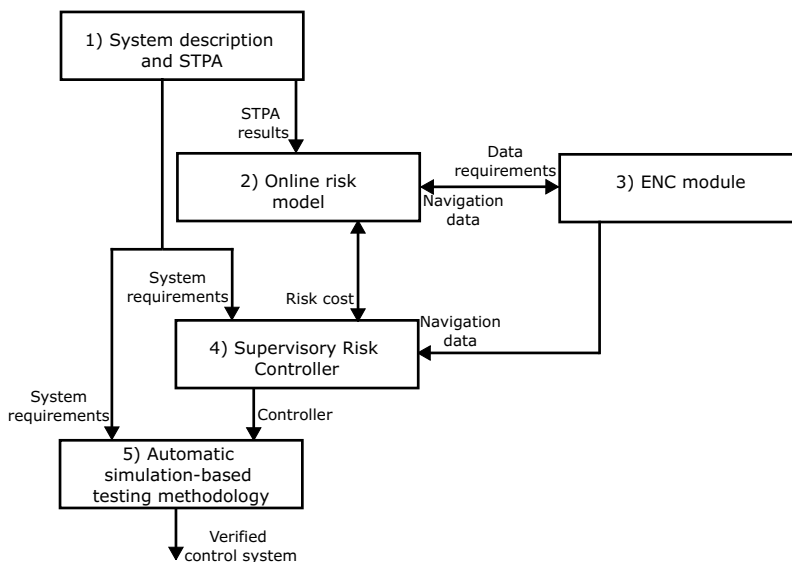


Figure 1: Methodology flowchart

2.1. Step 1: System description and STPA

To setup and build the control system, the ship and operation have to be described and analyzed, such as in terms of a CONOPS (concept of operations). This starts by clearly describing the ship, how it is controlled, its technical condition, and characterization of the operation that it is used for. In terms of control, it is important to know what type of controllers the ship has or will have, how they are connected, and their different responsibilities. Human operators or supervisors (e.g., onshore in a control center) must also be described with information about how they can control or affect the ship. Describing the ship's operation requires a clear statement of why and where the ship is sailing, as well as its operating modes. For example, a coastal cargo ship sailing along the Norwegian coast may be very different to a passenger ferry sailing between islands in the Mediterranean Sea.

The decisions or control actions relevant for the SRC must also be specified. These are important to consider because they are the only options for the SRC to affect the control of the ship. After describing the ship, STPA can be used to identify potential hazards, causal factors, and safety constraints. The STPA follows the steps defined in Leveson (2011) but is expanded to also explicitly consider the consequences of the hazardous events and system-level hazards as follows:

- a) Define the system
- b) Identify hazardous events and system-level hazards
- c) Identify unsafe control actions (UCAs)
- d) Develop loss scenarios
- e) Analyze consequences

The description of the ship can be used as a basis for the first step of STPA, and is a basis for defining the control structure and assigning responsibilities to the different controllers in the system. The next step is to identify hazardous events and to identify UCAs. These are subsequently described in loss scenarios that may lead to UCAs. Scenarios also include how decisions, such as selecting the wrong control mode or using machinery systems with failures, can lead to UCAs. The decisions are included in the same way as risk influencing factors (RIFs). The final part is to describe and classify the potential consequences of the hazardous events (e.g., through cost estimations).

2.2. Step 2: Online risk model

The online risk model is built based on the STPA results and follows the emerging top-down structure, like the results of the analysis, as shown in Figure 2. The BBN has six main types of nodes:

- Consequences
- Hazardous events
- System-level hazards
- UCAs
- RIFs
- Decisions

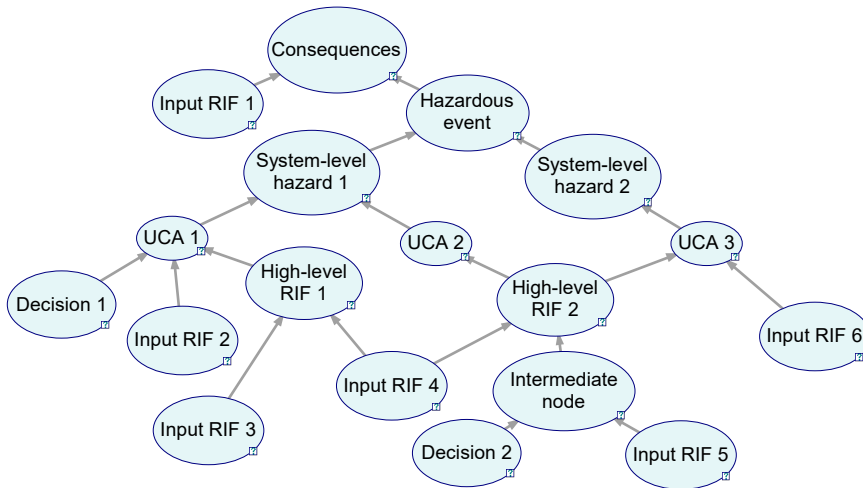


Figure 2: Example BBN structure, showing how the STPA is linked to the BBN and how different nodes are related (Adopted from Utne et al. (2020))

The end node in the BBN is the consequences. These are caused by the hazardous events, under given conditions. The hazardous events are caused by one or more system-level hazards identified in the STPA. The next is the UCAs that lead to system-level hazards. UCAs get an input from RIFs that describe the loss scenarios and the conditions where hazardous events have negative consequences. RIFs can be both high-level RIFs (H-RIFs) and input RIFs (I-RIFs), as shown in Figure 2. For a more detailed description of mapping STPA results to a BBN, the reader is referred to Utne et al. (2020) or Johansen and Utne (2022). For a detailed description of BBNs in general, the reader is referred to Fenton, N. and Neil, M. (2019).

The BBN is converted to an online risk model by deciding how to update the BBN as the ship sails with online information. This links specific nodes to sensors and systems onboard the ship, and then decides which data are necessary, including the ENC module. Decisions made in the SRC are also included in the BBN to model how they affect the risk picture and consequences. The BBN can also have intermediate nodes to group I-RIFs and decisions to reduce the number of nodes that are connected to each H-RIF. This is more important for larger and more complicated BBNs.

2.3. Step 3: ENC module

The ENC module extracts and manipulates data from electronic navigational charts. These data are necessary in the risk model to describe the surroundings and conditions around the ship. The ENC module is based on the open-source Python package SeaCharts (Blindheim and Johansen, 2022). This package uses FGDB 10.0 data sets with 2D data of the relevant areas. These are then processed as the application starts, so that they can be stored as shapefiles, where only the relevant depth layers and land areas are stored. This allows for much faster processing because it reduces the time necessary for computation and/or querying. The data is stored as polygons for various water depths and land areas. The stored shapefiles can then be queried to find the distance to points where the ship can collide or ground, and assess how much space the ship needs to maneuver.

The ENC module is set up by first loading the necessary maps for the relevant area. The next step is to define and load relevant layers for the ENC module, depending on the ship and data needed in the control system. This is achieved by defining the minimum water depth that the ship must maintain for safe sailing. To avoid unnecessary quantities of information in the risk model, a planning horizon is set in the ENC to decide how far the ENC should look ahead of the ship. This limits the data size that the ENC must query and reduces the computation time. Connecting the ENC module with the risk model is done by connecting the relevant nodes and updating them with data from the ENC, such as distance to land and shallow areas, combined with position and speed measurements from the GNSS system.

The current ENC module does not account for navigation markers, as this is not currently implemented in the SeaCharts package. This is discussed more in Section 4.5. For a detailed description of the package and all functions, the reader is referred to Blindheim and Johansen (2022).

2.4. Step 4: Supervisory risk controller

The controller is set up as an SRC to make high-level decisions or set control objectives. One option is to use costs as a means for implementing the inputs from the risk model into the decision making but there are other potential options, see Thieme et al. (2021).

For an autonomous ship controller, decisions can be made based on four costs: the risk cost from the online risk model, fuel cost based on the expected fuel consumption, operation costs (other than fuel), and the cost of not starting new missions. The total cost is calculated using Equation 1 as a function of the decisions, d :

$$C(d) = R(d) + F(d) + O(d) + L(d) \quad (1)$$

The risk cost, $R(d)$, gives the expected cost from the consequences described in the risk model. Fuel cost, $F(d)$, describes the expected cost of fuel of operating the ship under the current conditions. Operation cost, $O(d)$, describes the costs of operating the ship, outside of fuel cost, such as maintenance, insurance, and manning costs. $L(d)$ describes the potential loss of future income caused by the time used. The cost function is set up such that fuel cost, operation cost, and potential loss of future income increase if the ship takes a longer time to reach the final way-point.

The controller checks each possible set of decisions to find the set with the lowest cost. The decisions can vary depending on the ship and can include selecting what machinery mode to use, how the ship should be controlled, and which speed reference to follow. The SRC configures the control of the ship according to the set with the lowest cost.

2.5. Step 5: Automatic simulation-based testing methodology

Step five verifies the controller against a set of design requirements related to safety and efficiency. The verification process is performed using the automatic simulation-based testing methodology from Torben et al. (2022a). This methodology automatically runs simulations where the vessel is sailing along its planned route, while varying scenario parameters. The methodology formulates requirements using the Signal Temporal Logic (STL) formal specification language, which enables automatic evaluation of the simulations against the requirements (Maler and Nickovic, 2004). The result of evaluating a simulation against an STL requirement is an STL robustness score that describes how robustly the requirement is satisfied. If the STL score is greater than zero, then the requirement is satisfied. If it is less than zero, then the requirement is violated.

The methodology selects the simulations to run from a test space that is defined by a set of scenario parameters with corresponding parameter spaces. The test space can, for example, be based on scenarios that are identified in the STPA (Rokseth and Utne, 2019; Rokseth et al., 2018; Pedersen et al., 2022) to test the controller in specific situations. A Gaussian Process (GP) model (Rasmussen and Williams, 2006) is used to predict the STL robustness score as an unknown function of the test case parameters. The GP model estimates the expected value and the uncertainty of STL robustness over the entire parameter space of a test case. The GP model is iteratively updated by running simulations

and observing the resulting STL robustness score. The estimates of the GP model are then used to adaptively guide the test case selection towards cases with low STL robustness or high uncertainty. This results in efficient coverage of the parameter space or alternatively efficient falsification if the controller does not satisfy the requirements.

The testing terminates in a verified state if the lower confidence interval of the GP is greater than zero for the entire parameter space. For example, using 99% confidence intervals, a verification would indicate that there is at least a 99% probability that the system satisfies the requirement for the entire test space of the test case. Alternatively, if a test case that does not satisfy the requirements is identified, then the verification terminates in a falsified state, returning the corresponding counter-example. For a more detailed explanation of the automatic simulation-based testing methodology, the reader is referred to Torben et al. (2022a).

3. Case study: Supervisory risk control of an autonomous cargo ship

The method for building the SRC is tested in a case study that simulates an autonomous ship operating along the Norwegian coast to assess how the SRC manages and controls the ship in comparison to an existing conventionally-manned ship. The first part of the case study will analyze how the SRC adjusts the speed and configures the ship to maintain control. This is then compared performance-wise to a conventional ship in similar conditions, using position and speed data from the ship navigation system. The second part will study how the SRC handles failures in the machinery and propulsion system.

In the case study, it is assumed that the chart and GNSS measurements are sufficiently accurate to be used in the control system. It is also assumed that the time necessary to start up machinery can be neglected. There are still some delays and thruster dynamics included, such that engines and generators cannot change the load immediately. This is deemed sufficient to show how the SRC functions. Some of the potential ways to include these aspects in the SRC will be discussed in Section 4.3.

The ship simulation uses a simplified kinetic model without wave forces. This makes it easier to simulate and test the system, while it also changes the ship's movement such that the ship drifts more. This makes it more difficult to control the ship, especially in tight turns, without reducing the speed much more than conventional ships. Although the focus in this paper is the design and testing of the SRC, it still provides sufficient results to show that the proposed methodology works.

3.1. Step 1: Describing the ship and operation

The autonomous ship that is considered in the case study is an 80 m long and 16 m wide cargo ship that is sailing along the Norwegian coast. Although the ship is operated unmanned, it has a human supervisor onshore that can monitor and take control remotely if necessary. The ship has an autonomous control system, as shown in Figure 3, with an SRC as the high-level controller, an ANS to control the navigation, and an autonomous machinery management system (AMMS) to manage the machinery. The ANS has two ship operating (SO) modes: (i) DP and (ii) autopilot (AP), with a corresponding controller for each mode. The DP controller is used during low-speed maneuvering and station keeping, while the AP controller is used for transit at higher speeds. When the ship is operated in DP-mode, it utilizes the main propeller, bow tunnel thruster, and aft tunnel thruster to control the ship's speed, position, and heading. The AP controller uses the main propeller and rudder to control the ship.

The ship is equipped with a Liquefied Natural Gas (LNG) fueled main engine, a hybrid shaft generator (HSG), and two diesel generators. The HSG can be used as a generator to produce electricity when the main engine is used or an electric engine when diesel generators can be used to produce electricity.

The AMMS is used to control the machinery system depending on the machinery system operating (MSO) mode. The ship has three MSO-modes: power take out (PTO) mode, where the main engine provide propulsion and the HSG is used as a generator to provide electricity; power take in (PTI) mode, where the diesel generators produce electricity, and the HSG is used as an electrical engine to propel the ship; and the mechanical (Mech) mode is where the main engine provides propulsion and the diesel generators produce electricity.

The SRC is responsible for selecting SO-modes and MSO-modes. It also sets the reference speed for the ANS to follow.

The STPA in the case study is based on workshops with 13 relevant system experts who identified UCAs for the autonomous cargo ship. The participants have extensive experience working with risk assessment, testing, verification and validation, marine technology and maritime operation, and ship control system design in both academia and industry. The main purpose of the workshops was to not only identify how switching between different machinery

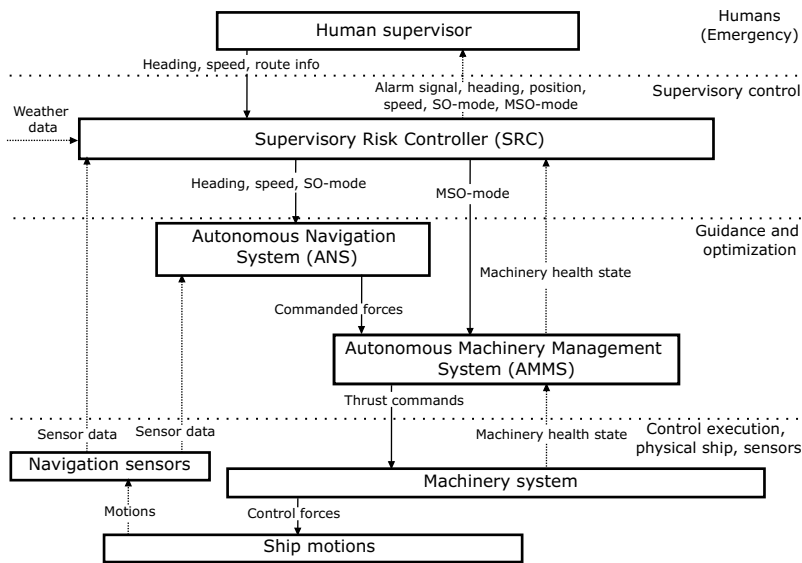


Figure 3: Hierarchical control structure (Adopted from Johansen and Utne (2022))

modes can lead to insufficient power capacity and power losses but also to identify when the wrong SO-mode used by the ANS could lead to accidents.

The STPA in the workshops considered a slightly different control structure with a remote operation center (ROC) that is responsible for planning, monitoring, and supervising the ship. The ANS and AMMS determine the SO- and MSO-mode, respectively, according to the sailing plan. An SRC in the control system was not included.

This case study assumes that the human supervisor plans the mission and the SRC then executes this plan. The human supervisor is also responsible for taking remote control of the ship if notified by the SRC. Selecting SO- and MSO-mode is now done by the SRC, and not the ANS and AMMS. The ANS controls the ship in either AP- or DP-mode depending on the SO-mode. The AMMS manages the machinery system according to the MSO-mode decided by the SRC. The AMMS also contains thrust allocation that computes individual thrust commands, based on the commanded forces from the ANS.

Since the workshops did not include an SRC, the control structure is modified to include this with the associated control actions. However, because setting SO-mode, MSO-mode, and the ship speed were considered when identifying UCAs in the workshops, the results can still be used in the case study.

The SRC has a set of process variables that are used to make decisions, as follows:

- PV-1: Active MSO-mode
- PV-2: Available power and thrust
- PV-3: Machinery system status
- PV-4: Active SO-mode
- PV-5: Ship's navigational states
- PV-6: Weather conditions
- PV-7: Traffic conditions
- PV-8: Route information

The case study focuses on the following hazardous event and system-level hazards, as follows:

- HE1: The ship grounds or has contact with the seafloor

- H1: The ship violates the minimum separation distance to the shore
- H2: The ship sails in water that is too shallow

The workshops identified a total of 60 UCAs. However, including all these would make the risk model more complicated to build and evaluate. Therefore, the case study focuses on five different UCAs, as shown in Table 1, to reduce the size and complexity of the risk model. These are chosen to have a good basis for specifying scenarios where the decision making in the SRC, such as setting SO-mode or speed reference, can lead to hazardous events and identify RIFs that affect this.

Table 1
Unsafe control actions

UCA	Description
UCA-1	A command is given to change MSO-mode to PTO when the health state of the ME is reduced
UCA-2	A command is given to change MSO-mode to Mech when the diesel generators do not function, or are unable to provide the rated power to the DC bus
UCA-3	A command is given to change MSO-mode to PTI, resulting in insufficient power for the main propulsion
UCA-4	A command is given to change SO-mode to transit/AP when the ship is in harbour/tight areas
UCA-5	A command is given to change SO-mode to maneuvering/DP when the speed is higher than the maximum maneuvering speed

Nine scenarios are defined to describe the situations that can cause UCAs and hazards, as presented in Table2.

Table 2
Scenarios

Scenario	Description	UCA
SC-1	MSO changed to PTO because PTI delivers insufficient amount of power but the health state of the ME is reduced, leading to insufficient power production	UCA-1
SC-2	MSO changed to PTO because the extra power in Mech is not necessary but the health state of the ME is reduced, leading to insufficient power production	UCA-1
SC-3	MSO changed to Mech because PTO is not producing sufficient power for propulsion but the diesel generators fail or provide less power than expected, leading to insufficient power on the DC bus	UCA-2
SC-4	MSO-mode is changed to from PTO to PTI due to an underestimate of the power necessary, leading to insufficient power to the ship	UCA-3
SC-5	MSO-mode is changed to from Mech to PTI due to an underestimate of the power necessary, leading to insufficient power to the ship	UCA-3
SC-6	SO-mode is changed to transit while still in harbor due to inaccurate/incorrect measurements of the ship states	UCA-4
SC-7	SO-mode is changed to transit while still in harbour due to wrong understanding of the area around the ship	UCA-4
SC-8	SO-mode is changed to maneuvering with too high speed due to faulty speed estimates/measurements	UCA-5
SC-9	SO-mode is changed to maneuvering with too high speed due to a wrong limit set in the controller	UCA-5

The extended STPA in this paper also considers the consequences from the hazardous event and the expected resulting costs. The consequences are divided into damage to own ship, damage to others' property, and harm to humans. Consequences are classified as either severe, significant, minor, or no consequences (IMO, 2018). Fatalities or serious injuries to humans or extensive damage to the ship or other ships/objects where assistance is necessary are considered severe consequences. Less serious/minor injuries to humans and damage that needs repairs outside of planned maintenance are considered significant consequences. Insignificant or no injuries to humans and damage that can be fixed in the next planned maintenance are considered minor consequences. Severe consequences cost 4 550 640 USD, significant 455 064 USD, minor 45 506.4 USD, and no consequences lead to zero cost. The costs are estimated based on EfficienSea (2012), The Norwegian Agency for Public and Financial Management (2018), and IMO (2018).

3.2. Step 2: Building the online risk model

The STPA is used as the basis to build the online risk model, as shown in Figure 4. The output from the risk model is the expected cost from the consequence. The BBN has four nodes describing the consequences: one general consequence node and one for damage to own ship, damage to others property, and harm to humans; one node describes the hazardous event, and one node describes each of the system-level hazards. The two system-level hazards depend on the five UCAs considered in the STPA. Each of these correspond to one node in the BBN.

The nine scenarios described in the STPA are used as the basis to define the six H-RIFs in the BBN. The list of H-RIFs, with the corresponding scenarios are show in Table 3. Each of the high-level RIFs are analyzed further to find I-RIFs, as shown in Table 4.

Table 3
Risk influencing factors

High-level RIF	Description	Scenario(s)
H-RIF-1	Machinery health state	SC-1,SC-2,SC-3
H-RIF-2	Estimation of necessary power	SC-1,SC-2,SC-3,SC-4,SC-5
H-RIF-3	Navigational complexity/situation	SC-1,SC-2,SC-3,SC-4,SC-5
H-RIF-4	Measurement/estimation of the ship's navigational states	SC-6, SC-8, SC-9
H-RIF-5	Situation awareness	SC-7, SC-8
H-RIF-6	Reliability of the ship's control system	SC-9

Table 4
Input to H-RIFs

High-level RIF	Description	Input RIF/Decision
H-RIF-1	Machinery health state	ME state, HSG state, DG1 state, DG2 state, BT state, AT state, MP state, ST state, MSO-mode (Decision node), SO-mode (Decision node)
H-RIF-2	Estimation of necessary power	PMS, AP performance/accuracy, DP performance/accuracy, SO-mode (Decision node)
H-RIF-3	Navigational complexity/situation	Traffic, Obstacles, Current, Distance to grounding hazard, Wind speed, Wind direction, SO-mode (Decision node) Speed reference (Decision node)
H-RIF-4	Measurement/estimation of ship's navigational states	GNSS system, Radar, AIS, SO-mode (Decision node) AP performance/accuracy DP performance/accuracy
H-RIF-5	Situation awareness	GNSS, Radar, AIS, Visual conditions
H-RIF-6	Reliability of the ship's control system	SO-mode (Decision node), AP performance/accuracy DP performance/accuracy, Ship design process

In addition to the I-RIFs and decisions in Table 4, the type of seabed and shore affect the consequences directly. Intermediate nodes are used between I-RIFs/decisions and H-RIF nodes to reduce the number of inputs to each node. This reduces the size of conditional probability tables (CPTs) and makes it easier to define these. CPTs and states are defined based on the work in Johansen and Utne (2022), DNVGL (2003), Hassel et al. (2021), discussions with crew working on different ships, and control engineers from Kongsberg Maritime. A full list of all nodes, with parent nodes, is shown in Table 5.

The BBN is converted to an online risk model by linking I-RIFs to the control system so they can be updated as the ship sails. Nodes describing the state of machinery parts are updated with information from the AMMS. If the machinery is well functioning and well maintained, then the probability of failure is very low, $9 \cdot 10^{-7}$. In future works, this is intended to be updated as the ship sails since machinery components are more likely to fail as components age, but this is not modeled in the current case study.

Nodes describing the control system and sensors are given a static value based on Johansen and Utne (2022), DNVGL (2003), Hassel et al. (2021). Weather nodes are linked to sensors where these exist, such as wind and current, or weather forecast and historical data (Norwegian Meteorological Institute, 2021). These nodes are designed to be

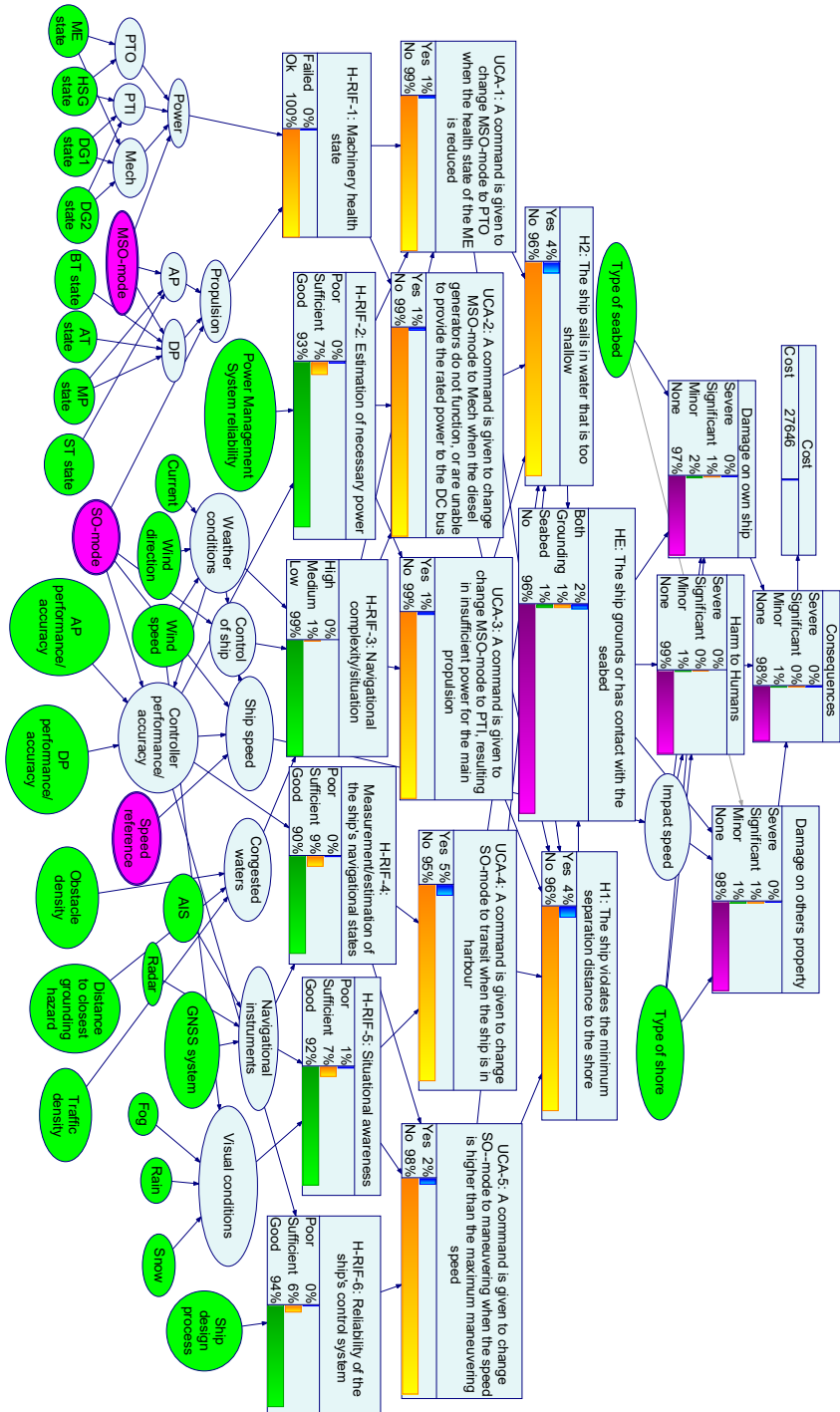


Figure 4: BBN risk model showing an example of the risk cost. For more detailed information about the BBN, please contact the corresponding author.

updated in real-time depending on the available data. Traffic use data is drawn from the automatic identification system (AIS), which is used to transmit the identity, position, course, and speed to nearby vessels using the very high frequency (VHF) band. Obstacle density and distance to grounding hazards are taken from the ENC. The seabed and shore are described with data from Norwegian Mapping Authority (2021) over the relevant area. The values used in input nodes describe the probability over the planned mission.

3.3. Step 3: Setting up the ENC module

The ENC module is setup to extract data from electronic navigational charts for use in the online risk model and the rest of the control system. The ENC module here includes charts covering the areas around Brønnøysund and Rørvik in Norway, which are relevant for the type of ship in the case study. The module is set up to consider everything shallower than 5 m as shallow areas or land where the ship cannot navigate safely. The rest of the chart is divided into layers of 10 m, 20 m, 50 m, 200 m, 350 m, and 500 m. This distribution is considered a reasonable combination of chart resolution and efficiency in the control system.

The obstacle density is based on the distance to the closest shallow point (i.e., areas with less than 5 m water depth) and the percentage of obstructed water around the ship. The water depth of 5 m is the same as the max draft of the ship. Using this water depth is considered sufficient for assessing the portion of obstructed water in this work. Shallow areas are then areas with too little water depth for the ship to sail which should be avoided with sufficient safety margins. The percentage of obstructed water is calculated by considering a disk with radius 1400 m and finding the portion of the disk with land and shallow water. The radius is set through testing to ensure that the disk gives a good picture of the sea area surrounding the ship, without being unnecessarily large.

The ENC module checks the area around the ship every 15 seconds and updates the input to the online risk model. Updating every 15 seconds ensure that the control system has updated data, while limiting the computation time necessary to check the ENC module.

3.4. Step 4: Building the supervisory risk controller

The SRC is the high-level controller that manages and controls the ship. The SRC uses data from the risk model and ENC, combined with operational measurements from the ANS and AMMS, such as position, speed, and machinery status to make decisions. The SRC has four main objectives: selecting the SO-mode, selecting the MSO-mode, setting the reference speed for the ship to follow, and notifying the human supervisor when the situation becomes too severe to continue.

The SRC is implemented as a switch that checks the cost function, as shown in Equation 1, for each set of decisions. The risk cost is calculated using Equation 2. This takes the probability of the different consequences multiplied with the cost for each consequence, as described in Section 3.1:

$$R(d) = Pr(severe)C_{severe} + Pr(significant)C_{significant} + Pr(minor)C_{minor} + Pr(none)C_{none} \quad (2)$$

The fuel cost is calculated as the specific fuel cost (SFC) multiplied by the expected sailing time. The SFC is taken from a look-up table, depending on wind speed, ship speed, current speed, and MSO-mode. The look-up table is made by simulating the machinery under different conditions to estimate how much fuel is used to sail a set distance. The fuel prices are taken from Ship & Bunker (2022) at 1 343,5 USD/ton for LNG and 684,5 USD/ton for diesel. This table provides a cost per distance that is multiplied with the planned sailing distance, as shown in Equation 3:

$$F(d) = SFC(wind, speed, current, machinery) * distance \quad (3)$$

Operation costs are calculated using Equation 4. This includes manning in the ROC, maintenance from wear and tear on the machinery, insurance of the ship, lubrication oil, spare-parts, and logistics. These are estimated based on conventional ships of the similar size and type, and using data from Stopford (2009) to be 341.3 USD/h for the current ship. This is similar to the fuel cost in normal transit with a speed of 5 – 7 m/s (9.7 – 13.6 knots):

$$O(d) = Cost_{operating} * distance/speed \quad (4)$$

The cost of potential future loss is calculated with Equation 5. This cost is the loss of income if the ship is unable to take on any new missions before finishing the current route, which is set to 910.1 USD/h:

$$L(d) = Cost_{futureloss} * distance/speed \quad (5)$$

The cost function, including the ratio between the different terms, is discussed in Section 4.7. The controller estimates the cost of sailing a distance equal to the initial route distance. This is constant for the whole route which keeps the weight between the different cost terms constant.

The alarm is implemented so that a human supervisor can take over control remotely of the ship if necessary, but unnecessary alarms also need to be avoided. To achieve an acceptable balance, the alarm trips if either the risk cost exceeds 9 267.70 USD, or the probability of the hazardous event exceeds 0.5. The cost limit is set between minor and significant consequences because it is better to have the human supervisor check the ship having an emergency later on. The SRC is implemented to lower the speed to limit the risk cost because impact speed directly affects the consequences. However, this can cause situations where the probability of a hazardous event is too high to continue due to environmental conditions, even though the risk cost is low because the speed is reduced to the minimum. Thus, a probability limit of 0.5 is used to notify the human supervisor in these situations.

If the SRC changes the ship's control configuration, then it is paused for 30 seconds before checking again. Implementing a time delay in the switching logic ensures that the controller reacts to changes but avoids situations where it gets stuck switching between different modes (e.g., DP and AP) without stabilizing, which is also called chattering (Utkin and Lee, 2006).

3.5. Step 5: Verifying the control system

After setting up the SRC, verification is done by first determining how to test the system and which requirements to verify against. The autonomous ship should follow the route through Brønnøysund that is shown in Figure 5. The route follows the same path as a conventional ship and those described in Norwegian Hydrographic Service (2018). This is used to check the ship in situations where the controller is expected to adjust the speed reference, without using much longer time than conventional ships. The ship has to lower the speed reference early enough to slow down when entering narrow and tight areas, and increase it when it opens up again.

To test safety, the ship should maintain a minimum distance of 5 m to shallow areas or provide an alarm to the human supervisor at least 5 min before the minimum distance is violated. Having a minimum distance of 5 m is not realistic for a real ship. However, to account for extra drift caused by simplifications in the simulator this is used to get results reasonable results that can be compared to conventional ships. These assumptions are discussed further in Section 4.8. The following verification focus on wind and how this affect the ship. However, the process is the same for other disturbances, such as current.

To verify that the controller is efficient, the ship should at maximum use 140 min on the whole route segment under consideration in the case study or provide an alarm to the human supervisor. This time limit is set based on the time existing manned ships used on the same route. Both the safety and efficiency requirements are tested in wind speeds ranging from no wind to 20 m/s and from all directions. Other factors (e.g., current, waves, and machinery failures) are not considered in the verification. This simplifies the verification but still gives sufficient results for further testing of the control system. The route is chosen to get a good variation between open water and more narrow straights with tight turns.

The verification is performed using the automatic simulation-based testing methodology that was introduced in Section 2.5. This methodology selects and simulates interesting combinations of wind speed and wind direction to verify or falsify the system. The system is verified to satisfy the safety requirement (minimum distance to shallow) in 161 simulations, and the efficiency requirement (maximum allowed sailing time) in 97 simulations. The STL robustness surfaces for safety and efficiency are shown in Figures 6a and 6b, respectively. The STL robustness score is normalized to the interval $[-1, 1]$. Figure 6a shows that the robustness score in the case study is always above 0. Similarly, Figure 6b shows that the robustness is always above 0 and is close to 1 when it reaches the final way-point early or trips an alarm because the risk cost or grounding probability becomes too high.

The verification shows that the control system makes the autonomous ship follow the route and it also reaches the end of the route in reasonable time in wind speeds of up to 8 m/s. Above this, the planned route forces the ship very close to land in certain spots, which means that it notifies the human supervisor. When the wind speed exceeds 10 m/s, the route leaves too little space for the ship to maneuver. This can cause problems with certain wind conditions. However, the control system provides an alarm to the human supervisor with enough time to pass the safety requirement. Overall, the verification shows that the proposed control system works in the planned route but it is limited by not being able to change the route in accordance with the environmental conditions.

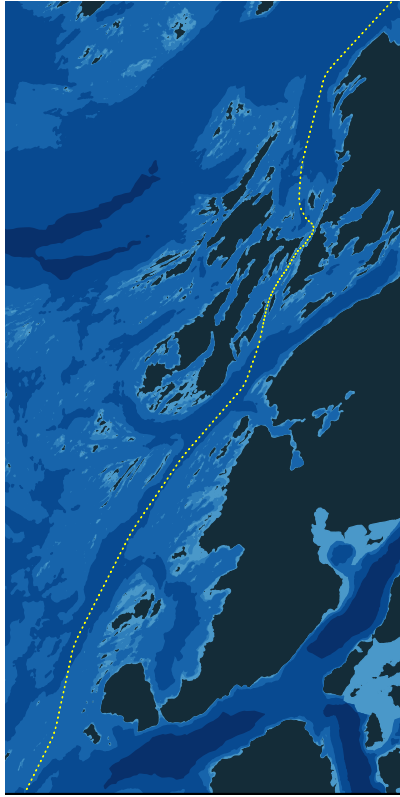


Figure 5: Route used in the verification process

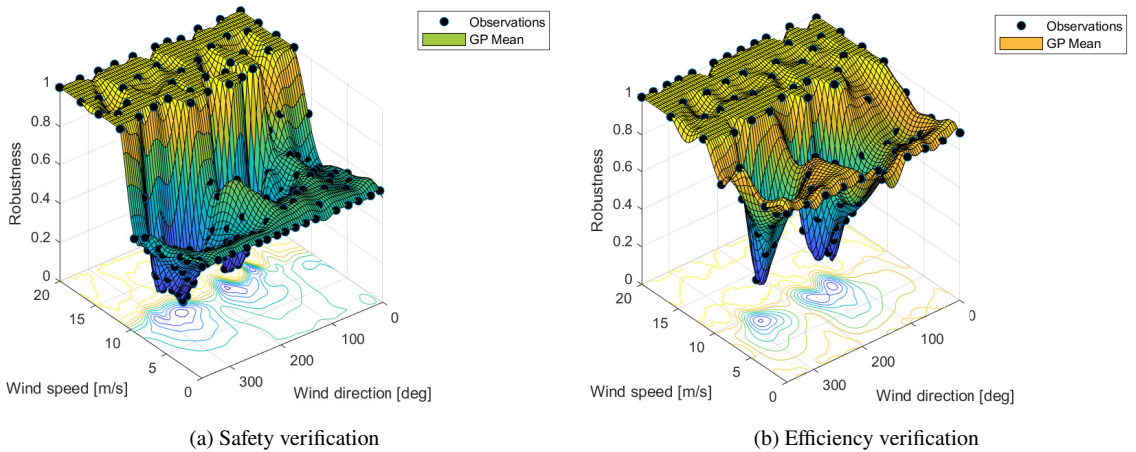


Figure 6: Robustness surfaces resulting from the two verification runs.

4. Results and discussion

4.1. Comparing the controller with the maneuvering of a conventional ship

After building and setting up the controller, the autonomous ship is simulated along two different routes to compare it against an existing conventional ship. The first route is through Rørvik and the second is through Brønnøysund. The

route through Brønnøysund is similar to the one used in the verification (Figure 5) but with different start and end points. The start and end points are changed because the GNSS data from the conventional ship is only available for part of the route. The purpose is to see how the SRC sets the speed reference, MSO-mode, and SO-mode, and compare this to how conventional ships operate along the same routes in similar weather conditions. The existing ship is equipped with a similar machinery and control system as the autonomous ship but with a crew who decides MOS-mode, SO-mode, and speed reference.

The conventional ship sailed through Rørvik and Brønnøysund in the fall of 2021 with a wind speed between 5-7 m/s. The routes followed by the conventional ship are plotted with GNSS data taken from the control system aboard the conventional ship. The route through Rørvik is planned by placing way-points along the route that the autonomous ship can follow. The GNSS data for Brønnøysund contain some measurements that place the route over land. The cause of these are not certain but it only affects the data between point 0.5 and 0.7. Therefore, the route was re-planned by placing way-points along the same route into Brønnøysund but following the route recommended in Norwegian Hydrographic Service (2018) through and after Brønnøysund. The routes are shown in Figures 7 for route one and 10 for route two with the conventional ship in red and the autonomous ship in yellow.

To compare the two ships, the risk model and SRC need position, speed, MSO-mode, and SO-mode from the conventional ship. Position and speed are recorded in the ship's control system. Ship speed is fed directly to the SRC to find the expected fuel cost and is used as input to the risk model. Position data is used in the ENC module to get the distance to the closest grounding hazard, and obstacle density. MSO-mode is set to PTO and SO-mode to AP after discussing how the conventional ship is operated with the crew. This provides a cost that can be compared to the autonomous ship. The SRC uses a constant distance when calculating costs, as explained in Section 3.4. The plots therefore show the costs of sailing a distance equal to the distance of the whole route, d_0 , estimated at each point.

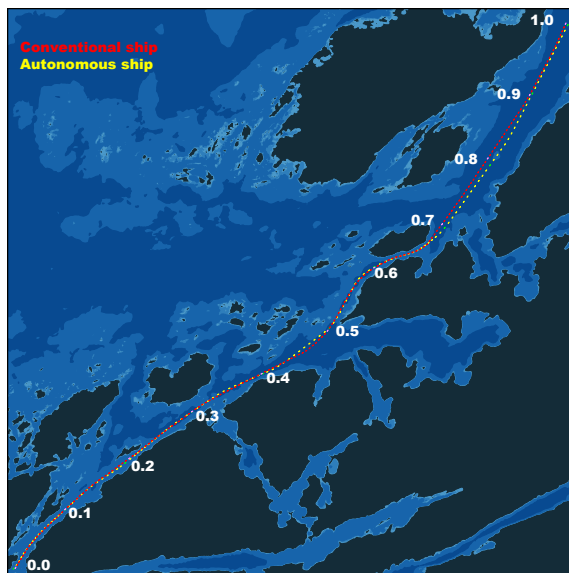


Figure 7: Map of route one through Rørvik. The conventional ship's route is shown in red and the autonomous ship's route is shown in yellow

4.1.1. Comparison on route one through Rørvik

On route one, the conventional ship starts with a speed of 5.25m/s, before increasing to 6.5m/s. The speed is then maintained at 6.5 – 6.75m/s the rest of the distance. The autonomous ship starts with a speed of 5m/s. This is later increased to 7m/s as the ship sails into more open water. Along the rest of the route, the speed varies between 5m/s and 7m/s as it passes through more narrow parts of the route and in more open areas. Overall, the autonomous ship varies the speed more as the environmental conditions change, compared to the conventional ship.

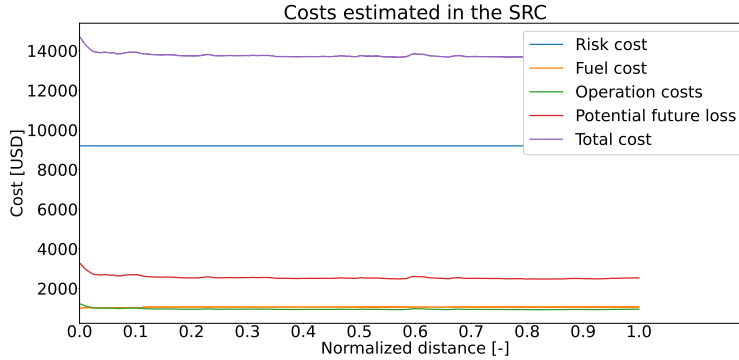


Figure 8: Conventional ship's costs on route one

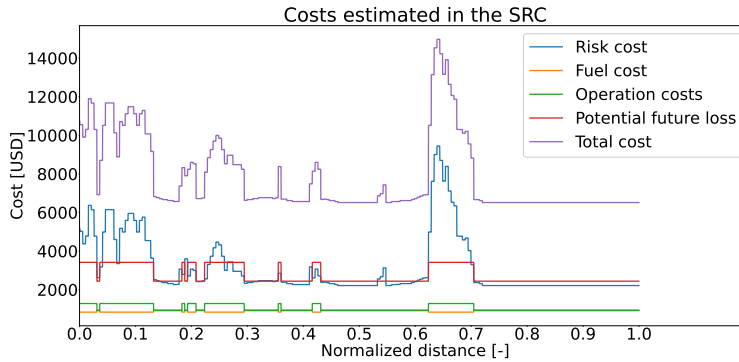


Figure 9: Autonomous ship's costs on route one

The cost is shown in Figure 8 for the conventional ship and in Figure 9 for the autonomous ship. The plots show the expected costs of sailing the full route, d_0 . The conventional ship has a higher risk cost (Blue line) because it maintains a higher minimum speed. Fuel (Yellow line), operation (Green line), and potential future loss (red line) costs are almost the same but they vary more for the autonomous ship because the expected time varies more corresponding to more changes in the speed. For the conventional ship, both fuel and operation costs are almost constant because the speed is kept more or less constant along the whole route. In contrast, the speed of the autonomous ship is changed more, which leads to more changes in fuel and operation costs. The conventional ship uses 96 minutes on the whole route and the autonomous ship uses 103 minutes.

4.1.2. Comparison on route two through Brønnøysund

The routes differ slightly more through Brønnøysund, due to the errors in the position data from the conventional ship. This means that the autonomous ship sails around 1 km longer. The conventional ship maintains a speed of around $6.75m/s$ before it reaches the narrow parts of the route. In the narrowest part, the speed is reduced to $3m/s$, it is then increased to $6.75 - 7m/s$ as the area opens up. The autonomous ship has a speed of $7m/s$ in open water. This is reduced to $5m/s$ when it reaches the first narrow straits. It then returns to $7m/s$ for a short time in the more open area, before it is reduced to $4m/s$ through the narrow harbour area. Overall, the autonomous ship makes more changes to the speed, but maintains a higher minimum speed.

The cost is shown in Figure 11 for the conventional ship and Figure 12 for the autonomous ship. Fuel (Yellow line), operation (Green line), and potential future loss (red line) costs are virtually the same along the whole route. The risk cost is similar along the first part but is much higher for the conventional ship in the middle part of the route. This is caused by the inaccuracies in the GNSS data that show the ship sailing over land. Fuel cost is similar for both ships

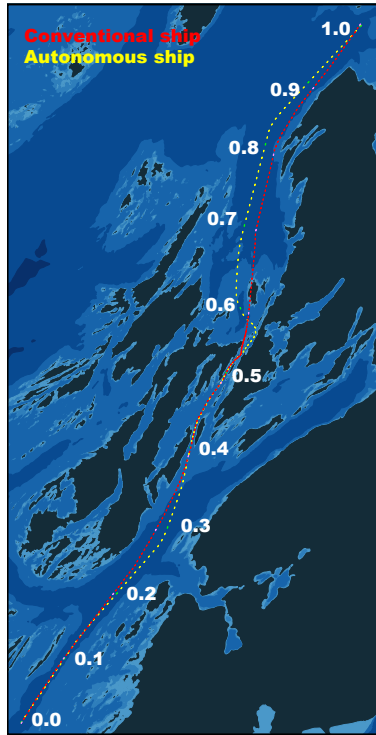


Figure 10: Map of route two through Brønnøysund. The conventional ship's route is shown in red and the autonomous ship's route is shown in yellow

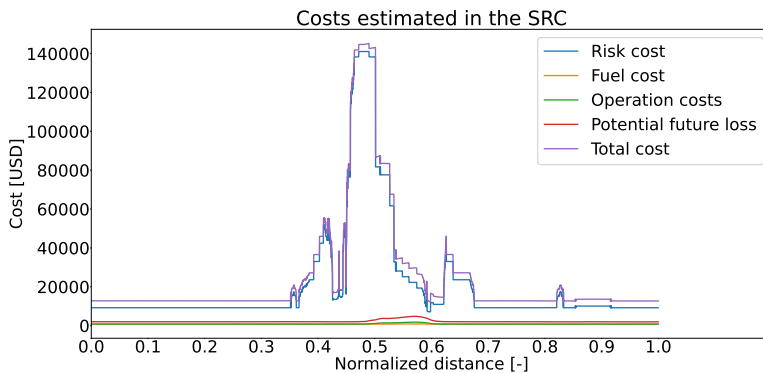


Figure 11: Conventional ship's costs on route two

with a reduced fuel consumption when the speed is reduced in the most challenging part of the route. Operation cost is also similar, but with a higher top for the conventional ship since because reduces the speed more.

4.2. Controlling the ship with machinery and propulsion failures

The second part of the case study tests how the control system manages the autonomous ship when the health of the main engine and steering system is worsened. This is modeled by increasing the probability of failure for these elements in the risk model. The SRC then chooses the best way to operate the ship based on this information. The

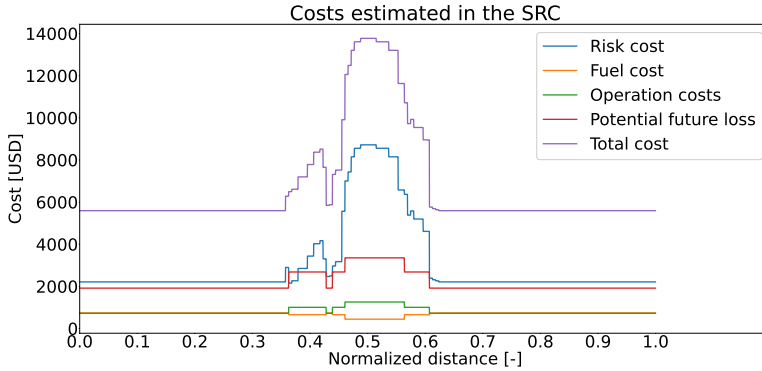


Figure 12: Autonomous ship's costs on route two

routes are the same as shown in Figure 7 for route one and Figure 10 for route two. The weather is also the same, which ensures that the results can be compared to how the ship is managed when all systems function.

4.2.1. Machinery and propulsion failures on route one through Rørвик

In both cases, the failure happens when the ship has sailed approximately 8 % of the route, close to point 0.1 on the figures. When the main engine fails, the SRC changes MSO-mode to PTI, which only uses the HSG and diesel generators for power production. The speed reference is also reduced to 4 m/s because the diesel generators produce less power than the main engine. This ensures that the ship still has sufficient power to maneuver. The SO-mode is AP along the whole route in this case.

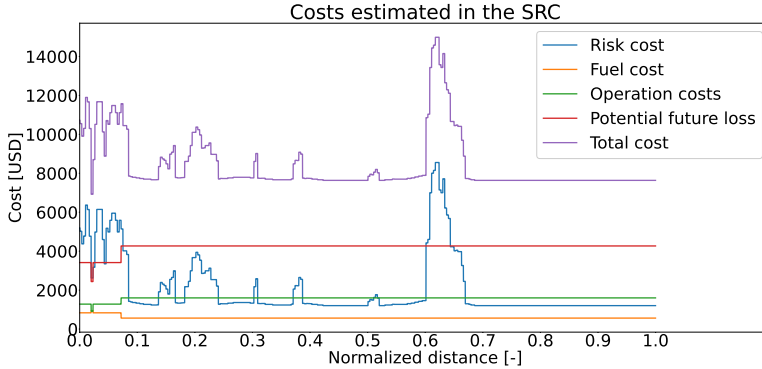


Figure 13: Costs with failure on main engine on route one

When the steering machinery fails, the speed is lowered significantly such that the tunnel thrusters can provide steering for the ship and SO-mode is changed to DP. The MSO-mode is Mech for the whole route. The speed reference switches between 2 m/s and 3 m/s, depending on the number of islands and obstacles around the ship.

Figures 13 and 14 show the costs calculated by the SRC. Overall, the cost is maintained at a similar level as when everything is working by adjusting how the speed is operated. The risk cost is controlled by reducing the speed, compared to how the ship is operated when all systems function as intended, and by switching to MSO-modes and SO-modes with functioning components. Operation and potential future loss is increased because the ship uses a longer time with lower speed.

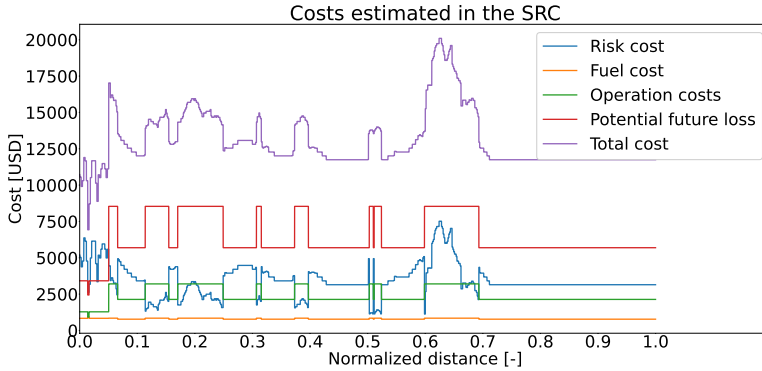


Figure 14: Costs with failure on steering machinery on route one

4.2.2. Machinery and propulsion failures on route two through Brønnøysund

The main engine fails between point 0.3 and 0.4, and the steering machinery fails between point 0.2 and 0.3. When the main engine fails, the speed is reduced significantly to account for the reduced power production. MSO-mode is also changed to PTL, which do not use the main engine. The SO-mode is AP along the whole route.

When the steering machinery fails, the speed is reduced to 2 m/s and SO-mode is changed to DP, to get more effect from the tunnel thrusters and maintain control of the ship. When the ship has passed the narrowest parts of the route, the speed is increased to 3 m/s.

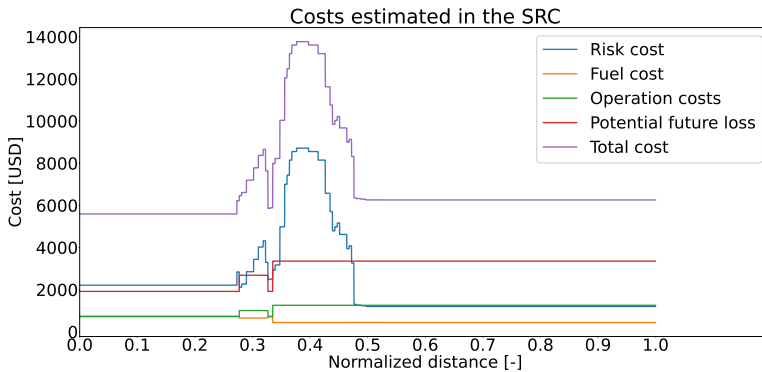


Figure 15: Costs with failure on main engine on route two

Similar to route one, the costs that are shown in Figure 15 for the main engine and Figure 16 are similar as when everything is functioning by reducing the speed and changing MSO-mode and SO-mode. The biggest difference compared to the cost when all systems function is the time used to finish the route. The time and the time dependent costs, operation costs and potential future loss increase when the ship sails at a lower speed. This is most visible after the ship has finished with the most challenging parts of the route, around 0.4-0.5. However, because the speed was reduced in the narrow and tight parts with all systems functioning as well, the max cost is still at the same level.

Data from conventional ships operating with failures but switching to modes that function without the failed components are limited, although this is a logical way to mitigate failures. In a conventional ship, the failed components can be fixed by the crew or the ship can be maneuvered to the closest harbor for repairs. On an autonomous ship without a crew, the only option is to maneuver to harbor and get it fixed there or in case of severe failures transport a repair crew to the ship offshore. Because this route change is not included in the SRC and the redundancy of the machinery systems onboard the autonomous ship was not compromised entirely, the ship continues to sail towards the final way-point. With

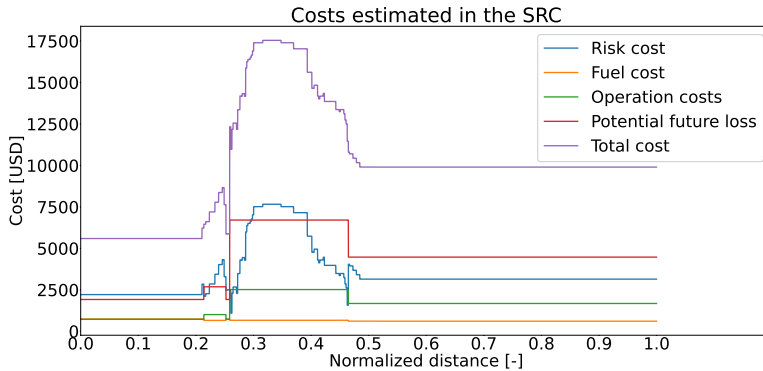


Figure 16: Costs with failure on steering machinery on route two

the current control system, this is a reasonable solution. Deviating from the planned route to get to shore and repair damaged equipment, which would be viable solutions in case of critical machinery failures and total loss of propulsion, and notifying the human supervisor are topics for further research that could improve the control system further.

4.3. Risk modelling and implementation in the control system

The proposed control system uses a BBN-based risk model to assess the risk. The model is based on an extended STPA of the ship. STPA provides a systematic way to analyze the ship and identify causal factors that can lead to hazardous events. The results of the STPA also provide a logical way to build and structure the BBN. However, the results depend on the data used and the quality of the analysis.

Another potential challenge using STPA is to decide the refinement level. The refinement level generally depends on the purpose of the STPA. More details mean more data, but it can also make the risk model and the corresponding calculations too time consuming. In this current work, the analysis considers one hazardous event only, two system level hazards, and five UCAs. The scenarios include causal factors, such as wind, obstacles, and the main parts of the machinery system. The scenarios could have been more detailed and could have included information about how machinery parts fail. However, because the purpose of the analysis in this paper is to build an SRC, the level of detail is considered to be sufficient because the controller does not provide detailed control actions to the different parts of the machinery systems. An example of this could be saying that the main engine can only produce limited power because the cooling system is only partially functioning, although in this situation limited power is necessary to maintain control of the ship. Enabling the controller to make such decisions would be an interesting topic for further research to continue to develop the control system.

When building the BBN risk model, the overall structure is determined by the STPA. However, because the SPTA is qualitative, it provides very little data for setting up states defining CPTs. Hence, they are generally based on other sources, such as literature, previous works, and expert judgement. The CPTs can also be adjusted later to put more weight on specific risk factors. Given that the CPTs are based on different sources, they contain a certain degree of uncertainty, as discussed in Section 4.7.

To convert the risk model into an online risk model, the risk model is connected to the rest of the control system. This means that all of the nodes in the BBN that can be measured by the control system or sensors should be updated when the ship is sailing. The risk model should be updated often to describe the current sailing conditions. However, updating it too often increases the computation time in the control system. There is also a limit to how quickly the controller can update the decisions. In the case study, the risk model and SRC is paused for 30 seconds if the SO-mode, MSO-mode, or speed reference is changed. This delay allows the controller to evaluate if the decisions influence the ship and to avoid chattering, where the controller is stuck switching back and forth between different decisions, such as DP and AP.

The control system can be expanded further by including more dynamics in the ship model. The case study assumes that machinery parts can be started immediately, which is not the case. Although the specific time necessarily varies for different engines, it will have to be included when making decisions. This type of dynamics could be included in the control system as limits to how often decisions can be changed. The risk model can also be modified to include

starters for the different machinery parts. For example, for the main engine to function, both the starter and engine would be necessary.

Similar dynamics can be included for changing load on the machinery and the speed of the ship. In particular, reducing the speed of the ship takes time, depending on the size of the ship. The ship simulator includes a time delay on load changes and uses some time to change the speed of the ship. However, the SRC does not account for this specifically when it makes decisions. Therefore, including more dynamics in the control system and risk model is an interesting topic for further research.

4.4. Challenges with measuring risk in cost function

The proposed control system uses a cost function to make decisions about MSO-mode, SO-mode, and speed reference. This cost function estimates the cost of operating and sailing the ship, and the potential cost of hazardous events. The cost of sailing and operating the ship is straightforward to calculate and use in a cost function because it is already measured as cost. However, to combine this with risk cost is a bigger challenge. The STPA analysis can identify potential hazardous events but is only a qualitative analysis that does not consider likelihood of these events or the following cost.

This work addresses this problem by extending the analysis to consider consequences and classifying these in terms of cost. The STPA results and consequences are modeled in a BBN to give a likelihood of the consequences. The likelihood is multiplied with the consequence cost to give a risk cost to use in the cost function. Decisions are then made based on the current time, without considering how this can change in the future. Risk could be alternatively assessed by simulating how changing conditions and decisions affect the cost over a longer time. This would make the SRC more like an MPC, which could find the optimum set of decisions to minimize the cost over a longer time period. However, this would mean running a lot of simulations to check all potential combinations. Investigating this further could be subject for further research.

4.5. Risk modeling and integration with the ENC module

In the proposed control system, information about grounding obstacles is important for the risk model because it allows the model to assess the area around the ship. This information, and other data about the relevant area, is available in ENCs. The ENC module is an efficient tool for extracting and filtering this information to enable it to be used to describe the navigation area in the risk model. The control system uses the distance to the closest area where the ship can ground and the density of such areas as inputs to the risk model. Together with weather and traffic data, this determines how challenging it is to maneuver the ship.

The ENC module used in this work do not account for navigational markers, as this is not currently implemented in SeaCharts. For an autonomous ship, knowing where different navigational markers and their meaning is an important part of operating safely. The proposed control system itself can utilize this information in the risk model to get a better understanding of the environment when this become available with the SeaCharts package. However, the current ENC module is still considered sufficient to demonstrate that the proposed control system works.

The ENC module also provides an efficient way to plot the ship during testing, and is used when testing the control system to see how well the ship follows the route and identifies problems in specific areas. Compared to just using the position data, without grounding obstacles and land, this approach makes it much easier to understand and/or verify how the ship maneuvers.

Data from the ENC module can also be used to add more functions to the control system, such as route planning. In addition, a planning algorithm can use the ENC module to check if the route maintains the necessary distance to land and grounding obstacles. When combined with AIS data, this can enable the planner to account for other ships and use this information to avoid collisions. This is an interesting extension of the control system that would reduce the need for human supervision and control even further. This point is left open as a relevant topic for further research.

4.6. The efficiency of testing and verification of control systems in operation

In this work, the proposed control system is verified against the design requirements using the automatic simulation-based testing framework that was introduced in Section 2.5. Using this approach significantly increases the efficiency of building sufficient verification evidence for the control system. Torben et al. (2022a) show that this reduces the number of simulations necessary to verify the scenario compared to a regular grid search, which is a large time saver when doing several design iterations and verifying the scenario after each iteration.

The robustness surface resulting from a verification run with the automatic testing framework enables us to quickly get an overview of the performance of the SRC system at different regions of the scenario space. This overview is actively used in the design process to iteratively adjust the control system. Compared to the alternative of running simulations manually and evaluating the resulting time series, this offers a significant reduction in the workload. Furthermore, using STL to evaluate the system also gives a robustness score to show not only that it is verified but also how well the system performs.

It is also worth noting that the verification process considers a specific route and area. These can be planned such that the route includes different environments, such as open water, coastal waters with many islands, or tight harbor areas. The results from the verification should then be valid for other routes with similar characteristics, as shown in the case study. However, if the system is only tested in a distinct environment, such as open water without obstacles, then it cannot say anything about how the controller handles other environments.

An interesting extension of the automatic testing framework is to also use it in an online setting and integrate it more closely with the SRC system. This online verification system could repeatedly start verification runs at fixed time intervals. A verification run would attempt to verify safe operation for a finite time-horizon ahead and for a set of uncertain scenario parameters, such as environmental conditions, traffic, or internal components failures. It would achieve this by running simulations with the current situation as an initial condition and then intelligently selecting the scenarios to simulate using the Gaussian process model. The simulator should have an exact (software-in-the-loop) replica of the SRC system, thereby also evaluating how future choices of the SRC system will affect the performance in the different scenarios. The result from a verification run would be used as a robustness map for future scenarios. This robustness map, when combined with data on the probability of the different scenarios, could then be used by the SRC system to make risk-based decisions. The concept of an online verification system operating in closed loop with the SRC system appears to be very interesting because it enables the SRC system to consider multiple future scenarios and at the same time evaluate how its decisions would affect future behavior.

Another interesting extension is to use the STPA directly to define safety requirements and simulation scenarios; see, for example, Rokseth et al. (2018); Rokseth and Utne (2019). In the current work, the scenarios are set up to test the ship in a wide range of wind conditions and in very different areas. However, testing similar scenarios to those that the STPA identified when controlling the ship is challenging. Therefore, testing in more specific scenarios based on the STPA is left for further research.

4.7. Uncertainties and sensitivity in the data and models in the case study

The proposed control system combines existing control systems, such as DP and autopilots, with an online risk model in an SRC. The DP and autopilot are well described in the literature and are used on conventional ships. However, the use of an online risk model in an autonomous ship system and the concept of a cargo ship sailing without humans onboard is a novel concept. This means that data describing this is very limited, and mostly based on concepts and plans for these types of ships.

To get sufficient data in the case study, a combination of data from traditional manned ships, concepts for autonomous ships, geographic, and weather data is used. The quality of geographical and weather data is good with little uncertainty. However, the case study considers a simplified environment and not all conditions that a real ship would experience. For example, the wind measurements are taken over a long period but only at a general location. The wind is therefore assumed to be the same along the whole route, even though it will likely vary significantly between different locations. Similarly, the charts that are used are the same as ships use for navigation today but are simplified to only consider shallow areas and land, and not other ships or navigational marks. Although these simplifications make it possible to test the system, they also lead to uncertainties in the results (e.g., how the system can handle more obstacles such as other ship traffic and more local variations in wind conditions).

The STPA used in the paper is based on a workshops with academic industry experts. This helps to identify relevant information for the case study but the quantitative risk models and corresponding calculations could still have limitations affecting the risk costs.

The input uncertainty will have a different effect on the overall uncertainty, depending on the sensitivity of each input node. If a node has high sensitivity, then changing it will change the risk cost more compared to nodes with lower sensitivity. Nodes with high sensitivity have the same effect on the uncertainty in the risk cost. Figure 17 shows the effect that each node has on the risk cost when setting the node in the best and worst state. This shows that the weather conditions have the biggest effect on the risk cost. Other input nodes with a significant effect on the cost are GNSS, machinery status, controller performance, and obstacles. The machinery and control system data are based on

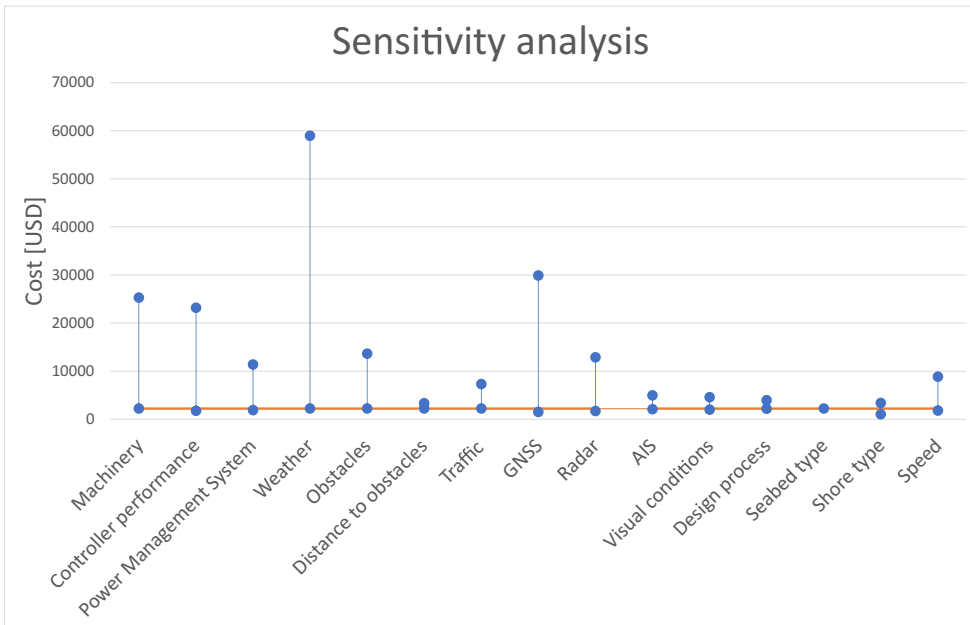


Figure 17: Sensitivity analysis, showing the effect on the risk cost of setting nodes in the best and worst states

multiple sources that describe the system’s reliability, and thus have less uncertainty. For weather and obstacles, the main source of uncertainty is the previously mentioned simplifications.

Another source of uncertainty in the risk model is the sensitivity of each input, or how much each input affect the risk cost. It is difficult to say how much weight should be on each input but it is possible to make some general remarks about it based on Figure 17. For an autonomous ship to function properly, it needs well-functioning machinery, power, and control system. It also makes sense that sensors providing situation awareness influence the ship, and that weather and obstacles affect the decision-making process. The sensitivity analysis and case study show that all these have a significant effect on the risk cost.

The fuel cost, operation cost, and loss of future income also affect the uncertainty in the case study. Because the SRC makes decisions based on the total cost, the balance between different cost elements affect the decisions and the results. The fuel cost is calculated using a lookup table of how much fuel the ship uses in different environmental conditions and speeds. The table is made by simulating the ship to derive the fuel consumption. These simulations use simplified models of the machinery system, but they still give numbers similar to those for existing ships and engines. Both operation costs and loss of future income are estimated based on the type of ship and operation.

Based on the tests, the balance between safety and efficiency is good. The balance between the different costs is also reasonable. Fuel and operation costs are at the same level. The potential loss of future income is slightly higher than the sum of fuel and operation costs because the ship should have a higher income than just covering the expenses. The results can be improved further by advancing the models, and by getting more and better data, but this is left for future work.

4.8. Simplifications in the ship simulator and testing

The proposed methodology and control system is tested using a simplified ship simulator. The simulator is based on the models given in Fossen (2011). This provides a good tool to test the ship’s control systems. However, the models include simplifications that affect the ship’s behavior and control. Not including wave forces is one such simplification. The most commonly used approach to include waves takes a 3D model of the ship and tests it in a hydrodynamic program. However, the data to make this 3D model is missing for the ship in the case study, and therefore the ship is simulated without waves. Similarly, the simulations consider a simplified propulsion system and use approximations in the kinematic and kinetic equations.

In testing, the simulator works sufficiently to test the proposed methodology and SRC. However, the ship is difficult to control when turning, especially using the autopilot. Therefore, the minimum distance used in the safety verification, Section 3.5, is only 5 m. In real life, the ship should stay further away from land. This would also add more safety margin to the ship draft and more clearance under the keel. Although the system has been tested with a larger minimum distance, it then fails the safety verification at much lower wind speeds. The ship can be operated in DP-mode, which offers much better control at lower speeds using the tunnel thrusters to both control heading and sideways position. However, this would mean sailing at unreasonable low speeds when compared to the conventional ship. To get comparable data, the autonomous ship is allowed to operate with smaller margins in the simulations. Given that the focus of the paper is the method for developing the SRC and how this make high level decisions, this is deemed sufficient. Testing with more accurate ship models is left for further work.

Accuracy in the position data is another challenge when testing the proposed methodology. The case study assumes that the GNSS data is accurate for use in the ship control system. However, GNSS accuracy can be a challenge for autonomous ships, especially when sailing between tall mountains where the signal quality can be affected by bad satellite coverage and signals reflecting off the mountains. How accurate the data is will vary depending on the location, but is something that should be addressed when setting the limits in the system verification and the control system. However, it is still sufficient for testing the SRC and the methodology for building this. Combining GNSS measurements with other sensors, such as radar, LIDAR, sonar, and cameras is an option for improving the accuracy by measuring the distance to land and other objects, instead of just using the GNSS position. However, this is considered to be outside the scope of this paper and is left for further work.

5. Conclusions

This paper presents a control system with risk-based decision-making capabilities to enable the smarter and safer operation of autonomous systems. The proposed control system uses an online risk model, which is represented by a BBN, to evaluate the operational risk, through an SRC. An ENC module is used to provide accurate data of the environment to both the risk model and the rest of the control system. The online risk model provides decision support in the SRC, which can make high level decisions. The control system has been verified against design requirements for safety (minimum distance) and efficiency (maximum time) using a novel formalized verification method. The combination of the SRC with ENC and formalized verification leads to a risk-based control system that can control autonomous ships in a safe and efficient manner, which currently does not exist.

The proposed control system is first compared to experimental data from an existing conventional ship in a case study along two coastal routes. This shows that the novel controller makes similar decisions to adjust the speed and maintain safe operation as the conventional ship, without using much more time to reach the end destination. The controller took slightly less risk than the conventional ship, mainly by adjusting the speed earlier when maneuvering in narrow areas, while maintaining a higher minimum speed than the conventional ship. The second part of the case study tests how the SRC handles failures in the machinery and propulsion system. This shows that the SRC changes MSO-mode and SO-mode to continue safely to the final way-point.

Further work includes adding more functions to the control system to increase autonomy, such as safe and reliable auto-docking. This will enable the ship to leave harbor, sail to a second location/harbor, deliver goods, and then return and dock in harbor again. This would be a typical cargo ship or passenger operation and would thus be an important step towards achieving highly autonomous ships.

Acknowledgments

The work by T. Johansen, S. Blindheim, T. Torben, I.B Utne, T.A. Johansen, and A.J. Sørensen is partly sponsored by the Research Council of Norway through the Centre of Excellence funding scheme, project number 223254, AMOS, and project ORCAS with project number 280655.

The authors would also thank Dr. Børge Rokseth at NTNU for his input and help with the simulation setup and in general discussions.

A. BBN connections

Tables with an overview of child/parent nodes in the BBN.

Table 5

BBN Nodes, Input-RIFs are only listed as parent nodes

Node description	Parent node(s)
Cost	Consequences
Consequences	Harm to humans, Damage on own ship, Damage on other ships/objects
Damage on other ships/objects	HE, Impact speed, Type of seabed, Type of shore
Damage on own ship	HE, Impact speed, Type of seabed, Type of shore
Harm to humans	HE, Impact speed, Type of shore
HE	H1, H2
H1	UCA-1, UCA-2, UCA-3, UCA-4, UCA-5
H2	UCA-1, UCA-2, UCA-3, UCA-4, UCA-5
UCA-1	H-RIF-1, H-RIF-2, H-RIF-3
UCA-2	H-RIF-1, H-RIF-2, H-RIF-3
UCA-3	H-RIF-2, H-RIF-3
UCA-4	H-RIF-4, H-RIF-5
UCA-5	H-RIF-4, H-RIF-5, H-RIF-6
H-RIF-1	Power, Propulsion
H-RIF-2	Power management system reliability, Controller performance/accuracy
H-RIF-3	Weather conditions, Control of ship, Congested waters
H-RIF-4	Controller performance/accuracy, Navigational instruments
H-RIF-5	Navigational instruments, Visual conditions
H-RIF-6	Controller performance/accuracy Ship design process
Power	PTO, PTI, Mech, MSO-mode
Propulsion	AP, DP
Weather conditions	Current, Wind direction, Wind speed
Control of ship	Weather conditions, SO-mode, Ship speed, Propulsion
Congested waters	Obstacle density, Distance to closest grounding hazard, Traffic density
Controller performance/accuracy	AP performance/accuracy, DP performance/accuracy, SO-mode, Weather conditions
Ship speed	Controller performance/accuracy, Speed reference
Navigational instruments	AIS, Radar, GNSS system
Visual conditions	Wind speed, Fog, Rain, Snow
PTO	ME state, HSG state
PTI	HSG state, DG1 state, DG2 state
Mech	ME state, DG1 state, DG2 state
AP	MSO-mode, MP state, ST state
DP	MSO-mode, MP state, BT state, AT state
Impact speed	Ship speed

CRedit authorship contribution statement

Thomas Johansen: Conceptualization, Methodology, Software, Investigation, Data Curation, Writing - original draft, Visualization. **Simon Blindheim:** Conceptualization, Software, Visualization, Writing - ENC methodology. **Tobias Rye Torben:** Conceptualization, Software, Investigation, Writing - Verification methodology. **Ingrid Bouwer Utne:** Conceptualization, Writing - review & editing, Supervision, Funding acquisition. **Tor Arne Johansen:** Conceptualization, Writing - review & editing, Supervision, Funding acquisition. **Asgeir J. Sørensen:** Conceptualization, Writing - review & editing, Supervision, Funding acquisition.

References

- Blindheim, S., Gros, S., Johansen, T.A., 2020. Risk-based model predictive control for autonomous ship emergency management. IFAC-PapersOnLine 53, 14524–14531.
- Blindheim, S., Johansen, T.A., 2022. Electronic navigational charts for visualization, simulation, and autonomous ship control. IEEE Access 10, 3716–3737.
- Bremnes, J.E., Norgren, P., Sørensen, A.J., Thieme, C.A., Utne, I.B., 2019. Intelligent risk-based under-ice altitude control for autonomous underwater vehicles. OCEANS 2019 MTS/IEEE Seattle , 1–8.

- Bremnes, J.E., Thieme, C.A., Sørensen, A.J., Utne, I.B., Norgren, P., 2020. A bayesian approach to supervisory risk control of AUVs applied to under-ice operations. *Marine Technology Society Journal* 54, 16–39.
- Brito, M., 2016. Uncertainty management during hybrid autonomous underwater vehicle missions. *Autonomous Underwater Vehicles 2016, AUV 2016*, 278–285.
- Brito, M., Griffiths, G., 2016. A bayesian approach for predicting risk of autonomous underwater vehicle loss during their missions. *Reliability Engineering & System Safety* 146, 55–67.
- Chaal, M., Valdez Banda, O.A., Glomsrud, J.A., Basnet, S., Hirdaris, S., Kujala, P., 2020. A framework to model the stpa hierarchical control structure of an autonomous ship. *Safety Science* 132.
- Chang, C.H., Kontovas, C., Yu, Q., Yang, Z., 2021. Risk assessment of the operations of maritime autonomous surface ships. *Reliability Engineering & System Safety* 207.
- DNVGL, 2003. DNV Report no 2003-0277 Annex II FSA 2003. Technical Report. DNVGL group technology & research. URL: <http://research.dnv.com/skj/FSALPS/ANNEXII.pdf>.
- EfficienSea, 2012. Methods to Quantify Maritime Accidents for Risk-based decision making. Technical Report. EfficienSea. URL: http://efficiensea.org/files/mainoutputs/wp6/d_wp6_4_1.pdf.
- Fan, C., Wröbel, K., Montewka, J., Gil, M., Wan, C., Zhang, D., 2020. A framework to identify factors influencing navigational risk for maritime autonomous surface ships. *Ocean Engineering* 202, 107188.
- Fenton, N. and Neil, M., 2019. *Risk Assessment and Decision Analysis with Bayesian Networks*. CRC Press.
- Fossen, T.I., 2011. *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley and Sons Ltd.
- Gil, M., 2021. A concept of critical safety area applicable for an obstacle-avoidance process for manned and autonomous ships. *Reliability Engineering & System Safety* 214.
- Hassel, M., Utne, I.B., Vinnem, J.E., 2021. An allision risk model for passing vessels and offshore oil and gas installations on the norwegian continental shelf. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 235, 17–32.
- Hu, L., Naeem, W., Rajabally, E., Watson, G., Mills, T., Bhuiyan, Z., Salter, I., 2017. Colregs-compliant path planning for autonomous surface vehicles: A multiobjective optimization approach. *IFAC-PapersOnLine* 50, 13662–13667.
- IMO, 2017. MSC.1/Circ.1580: Annex: Guidelines for vessels with dynamic positioning systems. IMO.
- IMO, 2018. Revised Guidelines for Formal Safety Assessment (FSA) for Use in the IMO Rule-Making Process. Technical Report. IMO. URL: <https://wwwcdn.imo.org/localresources/en/OurWork/Safety/Documents/MSC-MEPC%202-Circ%2012-Rev%202.pdf>.
- Johansen, T., Utne, I.B., 2020. Risk analysis of autonomous ships. e-proceedings of the 30th European Safety and Reliability Conference and 15th Probabilistic Safety Assessment and Management Conference (ESREL2020 PSAM15), 131–138.
- Johansen, T., Utne, I.B., 2022. Supervisory risk control of autonomous surface ships. *Ocean Engineering* 251, 111045.
- Leveson, N., 2011. *Engineering a Safer World: Systems Thinking Applied to Safety*. Engineering systems, MIT Press.
- Li, M., Mou, J., Chen, L., He, Y., Huang, Y., 2021. A rule-aware time-varying conflict risk measure for mass considering maritime practice. *Reliability Engineering & System Safety* 215.
- Loh, T.Y., Brito, M.P., Bose, N., Xu, J., Nikolova, N., Tenekedjiev, K., 2020a. A hybrid fuzzy system dynamics approach for risk analysis of auv operations. *Journal of Advanced Computational Intelligence and Intelligent Informatics* 24, 26 – 39.
- Loh, T.Y., Brito, M.P., Bose, N., Xu, J., Tenekedjiev, K., 2020b. Fuzzy system dynamics risk analysis (fusdra) of autonomous underwater vehicle operations in the antarctic. *Risk Analysis* 40, 818 – 841.
- Lyu, H., Yin, Y., 2019. COLREGS-Constrained Real-Time Path Planning for Autonomous Ships Using Modified Artificial Potential Fields. *Journal of Navigation* 72, 588–608.
- Maler, O., Nickovic, D., 2004. Monitoring temporal properties of continuous signals, in: *Lecture Notes in Computer Science*, pp. 152–166.
- Małka, J., Magaj, J., 2012. Data extraction from an electronic s-57 standard chart for navigational decision systems. *Proc. Zeszyty Naukowe/Akademia Morska Szczecinie*, 83–87.
- Norwegian Hydrographic Service, 2018. The Norwegian Pilot Guide. URL: <https://kartverket.no/en/at-sea/nautical-publications/the-norwegian-pilot-guide-sailing-directions>.
- Norwegian Mapping Authority, 2021. Norgeskart. URL: <https://norgeskart.no>.
- Norwegian Meteorological Institute, 2021. Met. URL: <https://www.met.no/en/weather-and-climate>.
- NSIA, 2019. Interim report 12 November 2019 on the investigation into the loss of propulsion and near grounding of Viking Sky, 23 March 2019. Technical Report. Norwegian Safety Investigation Authority. URL: <https://havarikommissjonen.no/Marine/Investigations/19-262>.
- Pedersen, T.A., Glomsrud, J.A., Ruud, E.L., Simonsen, A., Sandrib, J., Eriksen, B.O.H., 2020. Towards simulation-based verification of autonomous navigation systems. *Safety Science* 129, 104799.
- Pedersen, T.A., Neverlien, Å., Glomsrud, J.A., Ibrahim, I., Mo, S.M., Rindarøy, M., Torben, T., Rokseth, B., 2022. Evolution of safety in marine systems: From system-theoretic process analysis to automated test scenario generation. *International Conference on Maritime Autonomous Surface Ships*.
- Rasmussen, C.E., Williams, C.K.I., 2006. *Gaussian Processes for Machine Learning*. MIT Press.
- Rokseth, B., Utne, I.B., 2019. Deriving safety requirement hierarchies for families of maritime systems. *Transactions of the Royal Institution of Naval Architects Part A: International Journal of Maritime Engineering* 161, A229 – A243.
- Rokseth, B., Utne, I.B., Vinnem, J.E., 2018. Deriving verification objectives and scenarios for maritime systems using the systems-theoretic process analysis. *Reliability Engineering and System Safety* 169, 18 – 31.
- Sandøy, S.S., Hegde, J., Schjølberg, I., Utne, I.B., 2020. Polar map: A digital representation of closed structures for underwater robotic inspection. *Aquacultural Engineering* 89, 102039.
- Ship & Bunker, 2022. Rotterdam Bunker Prices. URL: <https://shipandbunker.com/prices/emea/nwe/nl-rtm-rotterdam#LSMG0>.
- Stopford, M., 2009. *Maritime economics*. 3rd ed. ed., Routledge.

- Tengesdal, T., Brekke, E.F., Johansen, T.A., 2020a. On collision risk assessment for autonomous ships using scenario-based mpc. IFAC-PapersOnLine 53, 14509–14516.
- Tengesdal, T., Johansen, T.A., Brekke, E., 2020b. Risk-based autonomous maritime collision avoidance considering obstacle intentions. Proceedings of 2020 23rd International Conference on Information Fusion, FUSION 2020 .
- The Norwegian Agency for Public and Financial Management, 2018. Guide in socio-economic analysis. URL: <https://dfo.no/fagomrader/utredning/samfunnsokonomiske-analyser/verdien-av-et-statistisk-liv-vsl>.
- Thieme, C.A., Rokseth, B., Utne, I.B., 2021. Risk-informed control systems for improved operational performance and decision-making. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability , 1748006X211043657.
- Torben, T., Glomsrud, J.A., Pedersen, T.A., Utne, I.B., Sørensen, A.J., 2022a. Automatic simulation-based testing of autonomous ships using gaussian processes and temporal logic. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability , 1748006X211069277.
- Torben, T., Smogeli, Ø., Utne, I.B., Sørensen, A.J., 2022b. On formal methods for design and verification of maritime autonomous surface ships. Proceedings of the World Maritime Technology Conference. .
- Utkin, V., Lee, H., 2006. Chattering problem in sliding mode control systems. IFAC Proceedings Volumes 39, 1. 2nd IFAC Conference on Analysis and Design of Hybrid Systems.
- Utne, I.B., Rokseth, B., Sørensen, A.J., Vinnem, J.E., 2020. Towards supervisory risk control of autonomous ships. Reliability Engineering & System Safety 196, 106757.
- Valdez Banda, O.A., Kannos, S., Goerlandt, F., van Gelder, P.H.A.J.M., Bergström, M., Kujala, P., 2019. A systemic hazard analysis and management process for the concept design phase of an autonomous vessel. Reliability Engineering & System Safety 191, 106584.
- de Vos, J., Hekkenberg, R.G., Valdez Banda, O.A., 2021. The impact of autonomous ships on safety at sea – a statistical analysis. Reliability Engineering & System Safety 210.
- Wang, H., Guo, F., Yao, H., He, S., Xu, X., 2019. Collision Avoidance Planning Method of USV Based on Improved Ant Colony Optimization Algorithm. IEEE Access 7, 52964–52975.
- Willners, J.S., Carreno, Y., Xu, S., Luczynski, T., Katagiri, S., Roe, J., Pairet, E., Petillot, Y., Wang, S., 2021. Robust underwater slam using autonomous relocalisation. IFAC-PapersOnLine 54, 273–280.
- Woo, J., Kim, N., 2020. Collision avoidance for an unmanned surface vehicle using deep reinforcement learning. Ocean Engineering 199, 107001.
- Wróbel, K., Montewka, J., Kujala, P., 2018. Towards the development of a system-theoretic model for safety assessment of autonomous merchant vessels. Reliability Engineering & System Safety 178, 209–224.
- Wróbel, K., Montewka, J., Kujala, P., 2017. Towards the assessment of potential impact of unmanned vessels on maritime transportation safety. Reliability Engineering & System Safety 165, 155 – 169.
- Xiao, G., Ren, B., Tong, C., Hong, X., 2021. A quantitative evaluation method for obstacle avoidance performance of unmanned ship. Journal of Marine Science and Engineering 9, 1127.
- Yin, J., Wang, Y., Lv, J., Ma, J., 2021. Study on underwater simultaneous localization and mapping based on different sensors. Proceedings of 2021 IEEE 10th Data Driven Control and Learning Systems Conference, DDCLS 2021 , 728–733.

Paper G:

Control Allocation for Double-ended Ferries with Full-scale Experimental Results

Tobias Rye Torben , Astrid H. Brodtkorb and Asgeir J. Sørensen

International Journal of Control, Automation and Systems

Volume 18, No. 3, pages 556–63

doi: *10.1007/s12555-019-0658-4*

Control Allocation for Double-ended Ferries with Full-scale Experimental Results

Tobias R. Torben* , Astrid H. Brodtkorb, and Asgeir J. Sørensen

Abstract: A novel control allocation algorithm for double-ended ferries with symmetrical thruster configuration is proposed. The allocation problem is formulated using the extended thrust representation, resulting in a four dimensional constrained optimization problem. Using the thrust configuration constraint, the optimization problem is reduced to a scalar bounded optimization problem, for which there exists fast solvers. We propose a cost function and bounds such that the allocation algorithm supports the standard way of performing manual thruster control on ferries. The real-time performance of the proposed algorithm is demonstrated in a simulation study, and in full-scale experiments.

Keywords: Autonomous ferries, control allocation, nonlinear optimization, thrust allocation.

1. INTRODUCTION

In the recent years there has been high activity related to autonomy in ferry operations, both in academia and in the industry [1–3]. Due to the relatively low mission complexity, ferry operations make a good candidate for piloting the transition towards increased autonomy in ships.

Automating the navigation tasks requires new developments for high-level control. However, at the control execution level, functionality resembling a traditional dynamic positioning (DP) system must exist. The performance and robustness of the DP system is paramount for the success of the mission. For over-actuated marine vessels, control allocation is a vital part of the DP system. Improper allocation may lead to degraded control performance, lower energy efficiency, and increased wear and tear on the actuators.

In this paper, we treat control allocation for double-ended ferries with symmetrical thruster configuration. This is a standard setup for car ferries, of which there exists several hundred in Scandinavia. A ferry crossing typically consists of a high-speed transit phase and a low-speed docking phase. During docking, fully actuated control is normally necessary to meet the high-precision requirements in all weather conditions. A control allocation algorithm is thus called for. These ferries have one azimuth thruster in each end with the same orientation, as

shown in Fig. 2. A challenge with this configuration is that it may take considerable time to change the direction of thrust, as the turning rate of the azimuths is usually low. In particular, when a braking force is required, one thruster must turn 180 degrees. This time delay is unacceptable for high-precision maneuvers, such as docking. Also, if not treated carefully, the thruster may produce a force in the wrong direction while turning, which may have a destabilizing effect on the motion control system.

For manual thruster control, it is common to turn the front thruster 180 degrees when approaching the dock. A force can then quickly be produced in both forward and reverse direction by balancing the thrust on the two thrusters. At some ferry sites, turning the front thruster by 180° is restricted, as the thruster wake may cause erosion damage to the quay. In this case the aft thruster is turned instead. The principle is the same, however some thruster-thruster interactions may occur.

As far as the authors are aware, no previous published work exists for control allocation for double-ended ferries. However, there is a rich literature in control allocation for marine surface vessels, commonly referred to as *thrust allocation*. In-depth reviews of the literature are given in [4] and [5]. Two methods dominate the literature. The Pseudo-inverse method [6], and variations of the Quadratic Programming (QP) method [7]. The Pseudo-inverse method has an advantage in its simplicity and low

Manuscript received August 13, 2019; revised November 3, 2019; accepted December 2, 2019. Recommended by Associate Editor Son-Cheol Yu under the direction of Editor-in-Chief Keum-Shik Hong. This work was supported by the Research Council of Norway through the Centres of Excellence funding scheme, project number 223254 - NTNU AMOS, and the KPN ORCAS project, project number 280655. The experimental testing was performed with funding from NTNU AMOS at the Autoferry project. Thanks to Brage Sæther for collaboration during testing.

Tobias R. Torben, Astrid H. Brodtkorb and Asgeir J. Sørensen are with the Department of Marine Technology, Norwegian University of Science and Technology (NTNU), Otto Nielsens vei 10, 7052 Trondheim, Norway (e-mails: {tobias.torben, astrid.h.brodtkorb, asgeir.sorensen}@ntnu.no).

* Corresponding author.



Fig. 1. The NTNU-developed, fully electric passenger ferry prototype, milliAmpere, used in the experimental testing. Photo: Kai T. Dragland.

computational complexity, but it yields an unconstrained solution. Also it does not support the thruster control method described for manual thruster control above. The strength of the QP method is that it can add both equality and inequality constraints. Drawbacks include relatively high computational complexity and the fact that the original allocation problem must be linearized before it can be formulated as a QP problem.

The main scientific contribution of this paper is the development of an efficient nonlinear control allocation algorithm. The algorithm uses the thrust configuration constraint to reduce the solution space, and is able to control the thrusters in a similar manner as described for manual thruster control. The algorithm is tested in simulation and in full-scale experiments with the passenger ferry prototype milliAmpere, shown in Fig. 1. The algorithm was originally developed for the NTNU Autoferry milliAmpere, and was first presented in [8]. Here it is elaborated upon with an extension for reversible thrusters and additional experimental results.

The paper is organized as follows: In Section 2 the problem formulation is described. In Section 3 the novel control allocation algorithm is presented. In Section 4 the results from the simulations and full-scale experiments are presented and discussed. Conclusions are given in Section 5.

2. PROBLEM FORMULATION

In this paper the control allocation problem for the thruster configuration shown in Fig. 2 is considered. This figure also shows the definition of the symbols and directions used in this paper.

For marine vessels with the horizontal plane (surge, sway, yaw) as its working space, the input to the thrust

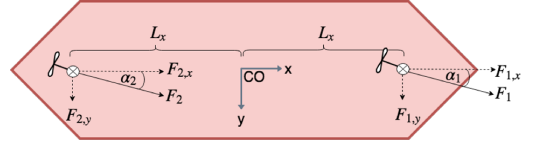


Fig. 2. Thruster configuration for double-ended ferries. F_i is the force from thruster i , with components $F_{i,x}$ and $F_{i,y}$ and angle α_i . L_x indicates the longitudinal distance from the center origin (CO) to each of the thrusters.

allocation module is the desired body frame control action $\tau = [X, Y, N]^T$. The output from the thrust allocation is the setpoints to the actuators. This can for instance be in form of propeller speed or pitch, engine torque or power, or rudder angle, depending on the type of actuator and the corresponding mapping of the desired force [9].

The mapping from actuator setpoints to body frame control action can be formulated as

$$\tau = B(\alpha)u, \quad (1)$$

where α is a vector of unknown actuator angles and u is an unknown vector of control inputs. The matrix B is called the *thrust configuration matrix*. The objective of the control allocation problem is to find an inverse mapping, that is, determine u and α such that the resulting generalized force produced is τ . For over-actuated marine vessels, the system of (1) is under-determined, that is, there are infinitely many solutions. This gives the thrust allocation algorithm freedom to choose a combination of u and α that is optimal in some sense.

For azimuth thrusters, the dependence of B on α can be removed by considering the surge and sway components of the thrust produced from one actuator. This is referred to as the *extended thrust representation* [6]. In the case of n azimuth thrusters, (1) takes the form

$$\tau = \begin{pmatrix} 1 & 0 & \dots & 1 & 0 & 0 \\ 0 & 1 & \dots & 0 & 1 & 0 \\ -L_{1,y} & L_{1,x} & \dots & -L_{n,y} & L_{n,x} & 0 \end{pmatrix} \begin{pmatrix} F_{1,x} \\ F_{1,y} \\ \vdots \\ F_{n,x} \\ F_{n,y} \end{pmatrix}, \quad (2)$$

where $L_{i,x}$ and $L_{i,y}$ are the distances from thruster i to the center origin (CO) in surge and sway directions, respectively. $F_{i,x}$ and $F_{i,y}$ are the surge and sway components of the thrust produced by thruster i . This is illustrated in Fig. 2 for $n = 2$ thrusters.

From the thrust components, the thrust and azimuth angle for each thruster can be retrieved as follows:

$$T_i = \sqrt{F_{i,x}^2 + F_{i,y}^2}, \quad \alpha_i = \arctan \frac{F_{i,y}}{F_{i,x}}. \quad (3)$$

3. NONLINEAR SCALAR CONTROL ALLOCATION

In this section the nonlinear scalar allocation (NSA) algorithm is presented, and guidelines for choosing the bounds on the optimization problem and the cost function are given. The section is concluded with a summary of the steps of the algorithm.

3.1. Transformation to a scalar, bounded optimization problem

When applying the extended thrust representation to the thruster configuration of Fig. 2, the thrust configuration matrix, $B \in \mathbb{R}^{3 \times 4}$, becomes

$$B = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ -L_{1,y} & L_{1,x} & -L_{2,y} & L_{2,x} \end{pmatrix}. \quad (4)$$

By assigning the index 1 to the front thruster and 2 to the aft thruster, and exploiting the symmetry properties of double ended ferries, it is clear that $L_{1,y} = L_{2,y} = 0$ and $L_{1,x} = L_x = -L_{2,x}$. Applying this to (4) yields

$$B = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & L_x & 0 & -L_x \end{pmatrix}. \quad (5)$$

A key observation here is that B is a Rank 3 matrix, whereas there are 4 unknown thrust components to be determined:

$$u = [F_{1,x}, F_{1,y}, F_{2,x}, F_{2,y}]^\top.$$

Hence, there is in fact only one degree of freedom in the thrust mapping $\tau = Bu$. The main idea for the new thrust allocation algorithm is to reformulate the original optimization problem with 4 variables (7, if slack variables are used as in [10]) into a bounded scalar optimization problem in the one, free variable of the equation $\tau = Bu$.

To do this reformulation, the structure of the solution space of $\tau = Bu$ is investigated. Firstly, the augmented matrix for the linear system is set up:

$$(B \mid \tau) = \begin{pmatrix} 1 & 0 & 1 & 0 & X \\ 0 & 1 & 0 & 1 & Y \\ 0 & L_x & 0 & -L_x & N \end{pmatrix}. \quad (6)$$

Secondly, Gaussian elimination is performed on $(B \mid \tau)$ until the matrix is in reduced row echolon form. This yields the equivalent linear system of equations

$$\begin{pmatrix} 1 & 0 & 1 & 0 & X \\ 0 & 1 & 0 & 0 & \frac{N+L_x Y}{2L_x} \\ 0 & 0 & 0 & 1 & -\frac{N-L_x Y}{2L_x} \end{pmatrix}. \quad (7)$$

Written in matrix-vector form, (7) becomes

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} F_{1,x} \\ F_{1,y} \\ F_{2,x} \\ F_{2,y} \end{pmatrix} = \begin{pmatrix} X \\ \frac{N+L_x Y}{2L_x} \\ -\frac{N-L_x Y}{2L_x} \end{pmatrix}. \quad (8)$$

Multiplying out (8) and writing out the components yields:

$$F_{1,x} + F_{2,x} = X, \quad (9a)$$

$$F_{1,y} = \frac{N + L_x Y}{2L_x}, \quad (9b)$$

$$F_{2,y} = -\frac{N - L_x Y}{2L_x}. \quad (9c)$$

This shows that $F_{1,y}$ and $F_{2,y}$ are uniquely determined, whereas in (9a) there is one degree of freedom. Natural choices for the parametrization of the solution space are $F_{1,x}$ or $F_{2,x}$. $F_{1,x}$ is chosen here.

For the next stage, the idea is to search for an optimal solution by trying different choices of $F_{1,x}$. For each step of the optimization, a candidate $F_{1,x}$ is selected. From this, $F_{2,x}$, $F_{1,y}$ and $F_{2,y}$ can be calculated from (9a) - (9c) such that the thrust configuration constraint is satisfied. Now that all the thrust components are known, the thrust magnitude and angle for each thruster can be calculated from (3). Knowing the thrust magnitude and angle for each thruster, a cost function can be defined to penalize, for instance, large thrust magnitudes or large changes of azimuth angle. This shows that the value of a cost function for all possible solutions to (9a) - (9c) can be calculated by only varying $F_{1,x}$. Two great advantages are thus achieved:

- 1) To search for the optimal solution, one only need to solve a *scalar* optimization problem.
- 2) For every candidate solution, the thrust configuration constraint, $\tau = Bu$, is automatically satisfied. This removes the need for equality constraints in the optimization problem, and the optimization problem can then be reduced to a *bounded* optimization problem, where the only constraints are fixed bounds on $F_{1,x}$.

The reason why these are great advantages, is that for scalar, bounded optimization problems, there exists fast and robust nonlinear solvers. Popular alternatives include Brent's Method [11] and Golden Section Search [12]. Two example implementations are MATLABs *fminbnd* and Python SciPys *fminbound*.

3.2. Choosing the bounds in the optimization problem

There are several options for choosing the bounds on $F_{1,x}$ in the optimization problem. First of all, the bounds should ensure that the allocation algorithm does not command a greater thrust than the thrusters can deliver. To ensure that a feasible solution exists for the optimization

problem, the commanded control action, $\tau = [X, Y, N]^\top$, should be saturated before entering the control allocation module. There are several options for doing this. The approach used in this paper is to first calculate $F_{1,y}$ and $F_{2,y}$ from (9b)-(9c), and saturate them according to the max thrust. X can then be saturated based on the remaining capacity in $F_{1,x}$ and $F_{2,x}$.

If it is desirable that the front is turned 180° , as described in Section 1, this can be achieved by constraining the front thruster to only produce a negative surge force, and the aft thruster to only produce a positive surge force.

To enforce these constraints, the requirement for the front thruster is that $F_{1,x} < 0$ and $F_{1,x} > -\sqrt{T_{max}^2 - F_{1,y}^2}$, where T_{max} is the maximum thrust produced by one thruster.

Similarly, the requirement for the aft thruster is that $F_{2,x} > 0$ and $F_{2,x} < \sqrt{T_{max}^2 - F_{2,y}^2}$. Since the variable of the optimization problem is $F_{1,x}$, these constraints must be expressed in terms $F_{1,x}$. This can easily be achieved using (9a). This gives that $F_{1,x} < X$ and $F_{1,x} > X - \sqrt{T_{max}^2 - F_{2,y}^2}$.

In the end, these constraints give two upper and two lower bounds on $F_{1,x}$. The bounds used in the optimization problem are chosen to be the most restrictive of the two. This yields the bounds

$$F_{1,x}^{min} = \max(-\sqrt{T_{max}^2 - F_{1,y}^2}, X - \sqrt{T_{max}^2 - F_{2,y}^2}), \quad (10)$$

$$F_{1,x}^{max} = \min(0, X). \quad (11)$$

3.3. Choosing a cost function

In optimal control allocation algorithms, it is common to penalize the thrust magnitude, to minimize energy consumption, and to penalize the change in thrust magnitude and the change in azimuth angle to reduce wear and tear [4]. As noted in Section 3.1, it is possible to evaluate these costs for all possible solutions by only varying $F_{1,x}$. Note that when the azimuth angles enter the cost function, it becomes nonlinear due to (3).

If the thruster control method where the front thruster is turned 180 degrees is used, the offset from the *home angles*, $\alpha_1 = 180^\circ$, $\alpha_2 = 0^\circ$, can also be penalized to avoid large or sudden changes in the azimuth angle. When tuning the weights in the cost function, there is a trade-off between energy efficiency and control performance. If less weight is enforced on the angle change and deviation from home angle and more weight is enforced on the thrust usage and thrust change, the algorithm will allow the thrusters to have larger angular displacements. For a given commanded sway force or yaw moment, a lower thrust is then needed to produce it, since the thrust will have a larger lateral component. However, due to the low servo speed, this will also yield a larger delay from commanded forces to produced forces.

In manual thruster control during docking, it is also common to give both thrusters a mean thrust opposing each other. This yields even faster response from commanded to produced control action, since it removes much of the spin-up time. Of course, it will also increase the energy consumption since the thrusters are constantly counteracting each other. The control allocation algorithm can easily be extended to support this by adding a term in the cost function which penalizes deviations from a prescribed mean thrust.

Using all the ideas introduced in this section, the cost function can take the form

$$C(F_{1,x}, \tau, \alpha^-, T^-) = w_T \|T\|^2 + w_{\Delta T} \|\Delta T\|^2 + w_{\delta T} \|\delta T\|^2 + w_{\delta\alpha} \|\delta\alpha\|^2 + w_{\Delta\alpha} \|\Delta\alpha\|^2, \quad (12)$$

where $\alpha^-, T^- \in \mathbb{R}^2$ are the azimuth angles and thrust magnitudes from last time step, and $T, \alpha \in \mathbb{R}^2$ are the thrust magnitudes and azimuth angles found from (3). $F_{2,x}$ is found from (9a). $\Delta\alpha \in \mathbb{R}^2$ are the shortest angle paths from α^- to α , $\delta\alpha \in \mathbb{R}^2$ are the shortest angle paths from α to the home angles. $\Delta T \in \mathbb{R}^2$ are the changes in thrust magnitude from last time step, and $\delta T \in \mathbb{R}^2$ are the deviations from the mean thrusts. $w_T, w_{\Delta T}, w_{\delta T}, w_{\Delta\alpha}, w_{\delta\alpha} \in \mathbb{R}_{\geq 0}$ are the corresponding weights.

Collecting all the penalized variables in a vector $z = [T, \Delta T, \delta T, \delta\alpha, \Delta\alpha]^\top \in \mathbb{R}^{10}$, the cost function can be written more familiarly as a quadratic form:

$$C(z) = z^\top Q z, \quad (13)$$

where $Q \in \mathbb{R}^{10 \times 10}$ is a positive semi-definite diagonal matrix of weights.

3.4. Thrusters with reversible thrust

The nonlinear scalar control allocation algorithm can easily be modified to support thrusters which can reverse the thrust. In this case, the thrusters do not need to turn 180 degrees to produce a braking force, and it is therefore not necessary to constrain the front thruster to only produce a negative force, and the aft thruster to produce a positive force. The constraints of (10)-(11) are thus modified to

$$F_{1,x}^{min} = \max(-\sqrt{T_{max}^2 - F_{1,y}^2}, X - \sqrt{T_{max}^2 - F_{2,y}^2}), \quad (14)$$

$$F_{1,x}^{max} = \min(\sqrt{T_{max}^2 - F_{1,y}^2}, X + \sqrt{T_{max}^2 - F_{2,y}^2}). \quad (15)$$

The solution of the optimization problem only gives the surge and sway components of the thrust for each thruster. However, when the thrusters can reverse the thrust, a given combination of $F_{i,x}$ and $F_{i,y}$ can be produced in two different ways, by reversing the thrust and turning the azimuth by 180 degrees. To find the optimal solution in this case, the optimization problem must be solved four times, with the following configurations:

- Forward thrust on both thrusters
- Forward thrust on front thruster, reversed thrust on aft thruster
- Reversed thrust on front thruster, forward thrust on aft thruster
- Reversed thrust on both thrusters

The solution with minimum cost of the four is chosen. Because most thrusters are less efficient in reverse, a term can be added to the cost function to penalize reverse thrusting. A solution with reversed thrust can still be favourable, because it may lead to significantly lower change in azimuth angle, and thus less control action delay.

3.5. Summary of the control allocation algorithm

To do one iteration of the nonlinear scalar allocation algorithm, the following steps must be performed:

- 1) Input the desired control action, τ , and the thrust magnitudes and angles from last time step. Saturate the desired control action to ensure that a feasible solution exists.
- 2) Set bounds on $F_{1,x}$ using, for instance, (10)-(11) or (14)-(15).
- 3) Calculate $F_{1,y}$ and $F_{2,y}$ from (9b) and (9c).
- 4) Formulate a cost function to minimize, for instance based on (13).
- 5) Solve a nonlinear bounded scalar optimization problem where $F_{1,x}$ is the free variable and $F_{1,y}$, $F_{2,y}$, τ are constant parameters.
- 6) Calculate $F_{2,x}$ from (9a).
- 7) Calculate the thrust magnitude and azimuth angle set-points from (3).

4. RESULTS AND DISCUSSION

To evaluate the performance of the novel control allocation algorithm, simulations and full-scale experiments are conducted. Simulations have the advantage that the produced thrust is known, and we can therefore compare the commanded and produced thrust. In experimental testing, the produced thrust is usually not known. Instead, the DP performance of the ferry when using the nonlinear scalar control allocation algorithm is evaluated, since good DP performance relies on good control allocation.

4.1. Simulation setup, results and discussion

In the simulation study, a simplified model for an azimuth thruster, which can not reverse the thrust, is used. The thruster dynamics are modelled as a saturated first order system:

$$\dot{T} = \frac{1}{\theta}(T_c - T), \quad (16a)$$

$$T = \max(0, \min(T_{max}, T)), \quad (16b)$$

where T_c is the commanded thrust, T is the actual thrust, T_{max} is the maximal thrust and θ is the thrust time constant.

The closed-loop azimuth servo is modelled as a proportional controller from angle offset to servo speed with saturation on the maximal servo speed. The dynamics of the servo is neglected, that is, the actual servo speed is assumed equal to the commanded servo speed.

$$\dot{\alpha} = r, \quad (17a)$$

$$r = \max(-r_{max}, \min(r_{max}, -K_p(\alpha - \alpha_c))), \quad (17b)$$

where α is the actual azimuth angle, α_c is the commanded azimuth angle, r is the servo speed, r_{max} is the maximal servo speed and K_p is the proportional gain.

The commanded control action is generated stochastically from discrete first-order Gauss-Markov processes [13]. The parameters of the stochastic processes are given in Table 1. The commanded control action has some noise, which is realistic for a signal from a DP controller.

The implementation of the control allocation algorithm uses all the terms from (13). The parameters are given in Table 1. For comparison, the Pseudo-inverse and QP methods are tested under the same conditions. The implementation of the QP method is that of [4], not including the singularity avoidance term.

Fig. 3 shows the commanded generalized force, τ together with the actual produced generalized force, Bu , from the nonlinear scalar allocation algorithm. The results show good tracking in all degrees of freedom. Fig. 4 shows the corresponding azimuth angles. The plot shows that both thrusters work in angles of $\pm 30^\circ$ about their home angles. There is good compliance between commanded and actual azimuth angles, indicating that the allocation generates feasible references for the azimuth servos.

Table 1. Parameters for simulations.

Description	Value
Thrust time constant	2.0 s
Max thrust	200 kN
Max servo speed	10 $\frac{\text{deg}}{\text{s}}$
Servo proportional gain	3.0 $\frac{1}{\text{s}}$
Distance from thruster to vessel center	50 m
Mean thrust	50 kN
Weight on angle change	10
Weight on thrust usage	0.1
Weight on thrust change	0.1
Weight on deviation from home angle	0.5
Weight on deviation from mean thrust	0.1
Gauss-Markov time constant	200 s
Gauss-Markov sample time	0.1 s
Gauss-Markov force standard deviation	10 kN
Gauss-Markov torque standard deviation	500 kNm

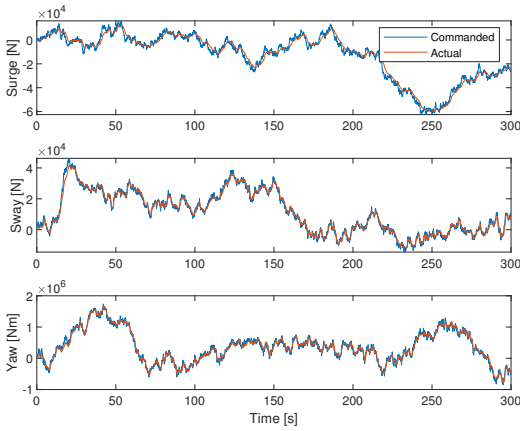


Fig. 3. Commanded and produced forces and moments for the nonlinear scalar control allocation algorithm.

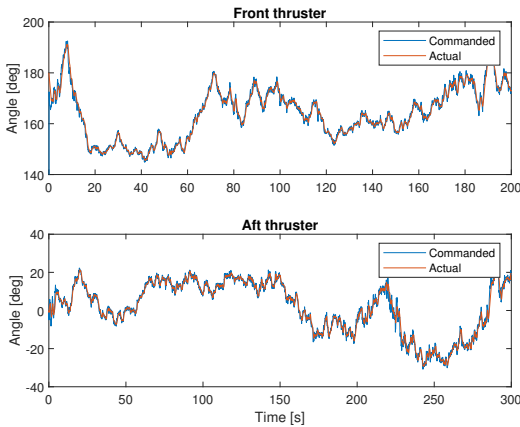


Fig. 4. Azimuth angles when using the nonlinear scalar control allocation algorithm.

Fig. 5 shows the cumulative error between commanded and actual produced generalized force for the three allocation methods; Pseudo-inverse, QP, and the nonlinear scalar allocation (NSA) algorithm proposed here. The figure indicates better performance for the nonlinear scalar allocation algorithm, although this can not be claimed on this basis alone, since there is a possibility of sub-optimal tuning for the QP method. It is believed that the main reason for the improved performance is due to penalizing large angular displacements. As discussed in Section 3.3 this gives less delay in the control action and thus tighter tracking and less accumulated error. As expected, this comes at the cost of higher thrust usage. In these simulations, the mean thrust was $40.0kN$ for the NSA method and $16.2kN$ for the QP method. The trade-off between performance and

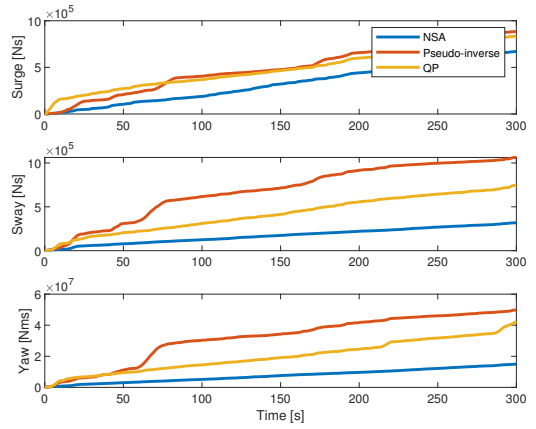


Fig. 5. Cumulative error for between commanded and produced generalized force for NSA, Pseudo-inverse and QP.

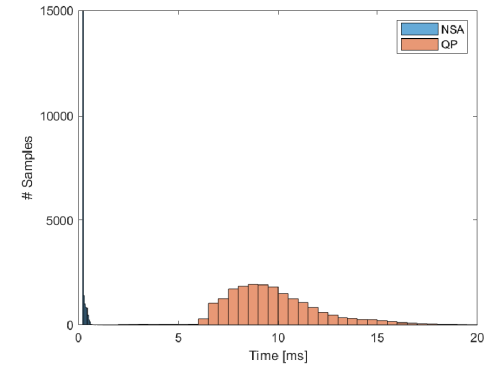


Fig. 6. Histogram of elapsed time for one iteration of NSA and QP for 20000 samples.

thrust usage is largely determined by the weight on deviation from home angles in the cost function. It should also be mentioned that an advantage of the QP method compared to the NSA method is better handling in the case that the commanded control action is greater than the propulsion system can produce because it can include slack variables in the optimization. The computational complexity of the QP method and the nonlinear scalar allocation algorithm are also compared. In the comparison, the MATLAB *fminbnd* solver was used for the nonlinear scalar algorithm and the MATLAB *quadprog* solver was used for the QP method. Fig. 6 shows histograms of elapsed time for one iteration. The nonlinear scalar allocation algorithm is, on average, 37.8 times faster. Also, it has a more narrow distribution. This is beneficial for robustness and predictability in a real-time control system, which is of particular

importance in autonomous vessels.

4.2. Full-scale experimental setup, results and discussion

The experimental tests were conducted with the passenger ferry prototype milliAmpere. A maneuvering model with hydrodynamic and rigid body data of this vessel can be found in [14]. The thrusters on this ferry can reverse the thrust, and the modifications presented in Section 3.4 was therefore used. We performed the commonly used 4-corner maneuver to evaluate the performance. The sides of the box are 10 meters, and the time used for one set-point change is about 40 seconds. Between the each set-point change, the ferry was performing stationkeeping for several minutes. In this way both the transient and stationkeeping DP performance was tested. The ferry starts in the lower-left corner and moves with the direction of the arrows. The maneuvers are becoming gradually more complex, as there is increasing coupling between the different degrees of freedom for each side of the square. The position and heading was obtained by Dual RTK GNSS, providing centimeter-level accuracy. The test was performed in calm conditions.

Fig. 7 shows the trajectory for the 4-corner DP test. The figure shows good DP performance both in the transient and stationkeeping phases. As expected, there are more deviations in the third and fourth maneuvers. This is likely due to coupling between the degrees of freedom and non-linear effects such a vortex shedding. However, compared to other 4-corner DP test results, these results are considered good. Fig. 8 shows ferry's the position and heading together with their respective references. Again, the results show good tracking. Some minor oscillations in the heading can be observed. The ferry has a flat keel, and therefore very little yaw damping. This is believed to be

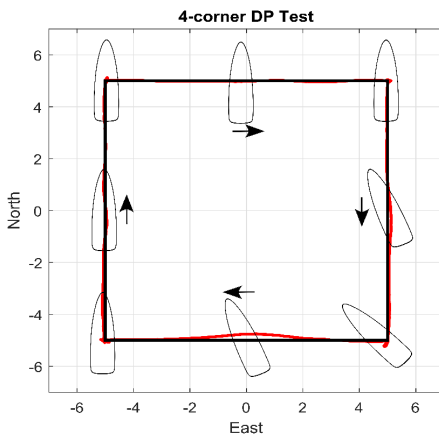


Fig. 7. Ferry trajectory from 4-corner DP test.

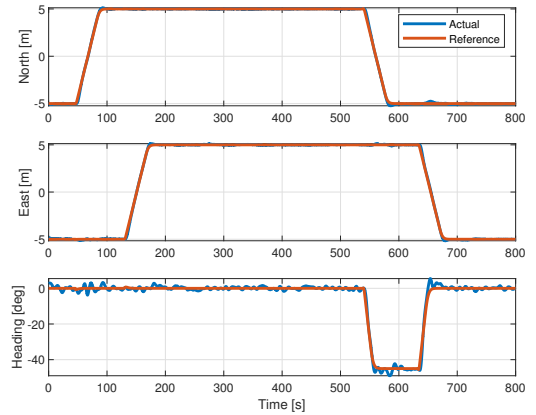


Fig. 8. Reference tracking from 4-corner DP test. Reference in red and actual in blue.

the reason for the heading oscillations. DP performance is highly dependent on the performance of the control allocation. These results therefore give increased confidence for the feasibility of the novel control allocation algorithm for use in real-time DP control systems.

5. CONCLUSION

A novel control allocation algorithm for double-ended ferries with symmetrical thruster configuration was presented. The algorithm reduces the dimension of solution space using the thrust configuration constraint, yielding a computationally efficient nonlinear optimization problem. Simulations and full-scale experimental results indicate promising real-time performance for use in a DP system.

REFERENCES

- [1] G. I. Bitar, *Towards the Development of Autonomous Ferries*, Master thesis, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, 2017.
- [2] Kongsberg Maritime, *Teknologi til fremtidens ferger - Kongsberg Gruppen*, <https://www.kongsberg.com/nb-no/kog/news/2018/april/teknologitilfremtidensferger/>, 2018.
- [3] Rolls-Royce Marine, *Press releases - Rolls-Royce and Finferries demonstrate world's first Fully Autonomous Ferry - Rolls-Royce*, <https://www.rolls-royce.com/media/press-releases/2018/03-12-2018-rr-and-finferries-demonstrate-worlds-first-fully-autonomous-ferry.aspx>, 2018.
- [4] T. I. Fossen and T. A. Johansen, "A survey of control allocation methods for ships and underwater vehicles," *Proc. of 14th Mediterranean Conference on Control and Automation, MED'06*, pp. 1-6, 2006.

- [5] T. A. Johansen and T. I. Fossen, "Control allocation - a survey," *Automatica*, vol. 49, no. 5, pp. 1087-1103, 2013.
- [6] O. J. Sjørdalen., "Optimal thrust allocation for marine vessels," *Control Engineering Practice*, vol. 5, no. 9, pp. 1223-1231, 1997.
- [7] E. Ruth, *Propulsion Control and Thrust Allocation on Marine Vessels*, Ph.D. thesis, NTNU, 2008.
- [8] T. R. Torben, "Control allocation for double-ended ferries with full-scale experimental results," *Proc. of 12th Control Applications for Marine Systems, CAMS 2019*, 2018.
- [9] Ø. Smogeli, E. Ruth, and A. J. Sørensen, "Experimental validation of power and torque thruster control," *Proceedings of the 20th IEEE International Symposium on Intelligent Control, ISIC '05 and the 13th Mediterranean Conference on Control and Automation, MED '05*, pp. 1506-1511, 2005.
- [10] T. A. Johansen, T. I. Fossen, and S. P. Berge, "Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 1, pp. 211-216, 2004.
- [11] R. P. Brent, "An algorithm with guaranteed convergence for finding a zero of a function," *The Computer Journal*, vol. 14, no. 4, pp. 422-425, 1971.
- [12] W. H. Press, S. A. Teukolsky, B. Flannery, and S. Teukolsky, *Numerical Recipes in C*, 1992.
- [13] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control*, Wiley, 2011.
- [14] A. P. Pedersen, *Optimization Based System Identification for the milliAmpere Ferry*, Master thesis, Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, 2019.



Tobias R. Torben received his M.Sc. degree in Marine Technology from NTNU in 2019, specializing in marine cybernetics. He is currently pursuing a Ph.D. in Marine Technology at NTNU. His research interests include simulation-based verification, formal methods and risk-aware control design, with applications to autonomous surface vessels.



Astrid H. Brodtkorb received her M.Sc. degree in Marine Technology at the Norwegian University of Science and Technology (NTNU) in Trondheim in 2014. She received her Ph.D. degree in Marine Control Systems at the Norwegian Centre of Excellence Autonomous Marine Operations and Systems (NTNU AMOS) in 2017. Her main research interests are hybrid control theory, observers, and sea state estimation applied to dynamic positioning (DP) systems for marine vessels.



Asgeir J. Sørensen obtained his M.Sc. degree in Marine Technology in 1988 at NTNU, and a Ph.D. degree in Engineering Cybernetics at NTNU in 1993. He has more than 15 years of industrial experience from ABB and Marine Cybernetics (DNV GL). Since 1999, he has held the position of Professor of Marine Control Systems at the Department of Marine Technology, NTNU. He is currently acting as a key scientist and the Director of the Centre for Autonomous Marine Operations and Systems (NTNU AMOS).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**PhD Theses Published at the
Department of Marine Technology**

**Previous PhD theses published at the Department 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-79- x	Finn Gunnar Nielsen, MH	Hydrodynamic problems related to oil barriers for offshore application
UR-80-06	Nils Sandmark, 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-	Langhelle, Nina K., MK	Experimental Validation and Calibration of

128		Nonlinear Finite Element Models for Use in Design of Aluminium Structures Exposed to Fire. (Dr.Ing. Thesis)
MTA-99-129	Berstad, Are J., MK	Calculation of Fatigue Damage in Ship Structures. (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, Qinzhen, MH	Wash and wave resistance of ships in finite water depth. (Dr.Ing. Thesis)
MTA-2002-154	Melhus, Øyvinn, 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	Chezian, 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-2007-26	Aronsen, Kristoffer Høy	An experimental investigation of in-line and 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	Grytøy, 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 and 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 using 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 Skewed Three-Dimensional Channel Flow, Ph.d.-thesis, IMT.
IMT 2009-56	Ayala-Uraga, Efrén	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 Intact and Damaged Ships in a Collision 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 Particle 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 – 2011-77	Amini, Hamid	Azimuth Propulsors in Off-design Conditions. 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 engines 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	Ramirez, Pedro Agustin Pérez	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 level 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	Böckmann, 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 Jesús 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 efficiency 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 Simulation 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 efficiency 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 Bruslerud	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
IMT-1-2018	Yingguang Chu	Virtual Prototyping for Marine Crane Design and Operations
IMT-2-2018	Sergey Gavrilin	Validation of ship manoeuvring simulation models
IMT-3-2018	Jeevith Hegde	Tools and methods to manage risk in autonomous subsea inspection, maintenance and repair operations
IMT-4-2018	Ida M. Strand	Sea Loads on Closed Flexible Fish Cages
IMT-5-2018	Erlend Kvinge Jørgensen	Navigation and Control of Underwater Robotic Vehicles

IMT-6-2018	Bård Stovner	Aided Inertial Navigation of Underwater Vehicles
IMT-7-2018	Erlend Liavåg Grotle	Thermodynamic Response Enhanced by Sloshing in Marine LNG Fuel Tanks
IMT-8-2018	Børge Rokseth	Safety and Verification of Advanced Maritime Vessels
IMT-9-2018	Jan Vidar Ulveseter	Advances in Semi-Empirical Time Domain Modelling of Vortex-Induced Vibrations
IMT-10-2018	Chenyu Luan	Design and analysis for a steel braceless semi-submersible hull for supporting a 5-MW horizontal axis wind turbine
IMT-11-2018	Carl Fredrik Rehn	Ship Design under Uncertainty
IMT-12-2018	Øyvind Ødegård	Towards Autonomous Operations and Systems in Marine Archaeology
IMT-13-2018	Stein Melvær Nornes	Guidance and Control of Marine Robotics for Ocean Mapping and Monitoring
IMT-14-2018	Petter Norgren	Autonomous Underwater Vehicles in Arctic Marine Operations: Arctic marine research and ice monitoring
IMT-15-2018	Minjoo Choi	Modular Adaptable Ship Design for Handling Uncertainty in the Future Operating Context
MT-16-2018	Ole Alexander Eidsvik	Dynamics of Remotely Operated Underwater Vehicle Systems
IMT-17-2018	Mahdi Ghane	Fault Diagnosis of Floating Wind Turbine Drivetrain- Methodologies and Applications
IMT-18-2018	Christoph Alexander Thieme	Risk Analysis and Modelling of Autonomous Marine Systems
IMT-19-2018	Yugao Shen	Operational limits for floating-collar fish farms in waves and current, without and with well-boat presence
IMT-20-2018	Tianjiao Dai	Investigations of Shear Interaction and Stresses in Flexible Pipes and Umbilicals
IMT-21-2018	Sigurd Solheim Pettersen	Resilience by Latent Capabilities in Marine Systems
IMT-22-2018	Thomas Sauder	Fidelity of Cyber-physical Empirical Methods. Application to the Active Truncation of Slender Marine Structures
IMT-23-2018	Jan-Tore Horn	Statistical and Modelling Uncertainties in the Design of Offshore Wind Turbines

IMT-24-2018	Anna Swider	Data Mining Methods for the Analysis of Power Systems of Vessels
IMT-1-2019	Zhao He	Hydrodynamic study of a moored fish farming cage with fish influence
IMT-2-2019	Isar Ghamari	Numerical and Experimental Study on the Ship Parametric Roll Resonance and the Effect of Anti-Roll Tank
IMT-3-2019	Håkon Strandenes	Turbulent Flow Simulations at Higher Reynolds Numbers
IMT-4-2019	Siri Mariane Holen	Safety in Norwegian Fish Farming – Concepts and Methods for Improvement
IMT-5-2019	Ping Fu	Reliability Analysis of Wake-Induced Riser Collision
IMT-6-2019	Vladimir Krivopolianskii	Experimental Investigation of Injection and Combustion Processes in Marine Gas Engines using Constant Volume Rig
IMT-7-2019	Anna Maria Kozłowska	Hydrodynamic Loads on Marine Propellers Subject to Ventilation and out of Water Condition.
IMT-8-2019	Hans-Martin Heyn	Motion Sensing on Vessels Operating in Sea Ice: A Local Ice Monitoring System for Transit and Stationkeeping Operations under the Influence of Sea Ice
IMT-9-2019	Stefan Vilsen	Method for Real-Time Hybrid Model Testing of Ocean Structures – Case on Slender Marine Systems
IMT-10-2019	Finn-Christian W. Hanssen	Non-Linear Wave-Body Interaction in Severe Waves
IMT-11-2019	Trygve Olav Fossum	Adaptive Sampling for Marine Robotics
IMT-12-2019	Jorgen Bremnes Nielsen	Modeling and Simulation for Design Evaluation
IMT-13-2019	Yuna Zhao	Numerical modelling and dynamic analysis of offshore wind turbine blade installation
IMT-14-2019	Daniela Myland	Experimental and Theoretical Investigations on the Ship Resistance in Level Ice
IMT-15-2019	Zhengru Ren	Advanced control algorithms to support automated offshore wind turbine installation
IMT-16-2019	Drazen Polic	Ice-propeller impact analysis using an inverse propulsion machinery simulation approach
IMT-17-2019	Endre Sandvik	Sea passage scenario simulation for ship system performance evaluation

IMT-18-2019	Loup Suja-Thauvin	Response of Monopile Wind Turbines to Higher Order Wave Loads
IMT-19-2019	Emil Smilden	Structural control of offshore wind turbines – Increasing the role of control design in offshore wind farm development
IMT-20-2019	Aleksandar-Sasa Milakovic	On equivalent ice thickness and machine learning in ship ice transit simulations
IMT-1-2020	Amrit Shankar Verma	Modelling, Analysis and Response-based Operability Assessment of Offshore Wind Turbine Blade Installation with Emphasis on Impact Damages
IMT-2-2020	Bent Oddvar Arnesen Haugalokken	Autonomous Technology for Inspection, Maintenance and Repair Operations in the Norwegian Aquaculture
IMT-3-2020	Seongpil Cho	Model-based fault detection and diagnosis of a blade pitch system in floating wind turbines
IMT-4-2020	Jose Jorge Garcia Agis	Effectiveness in Decision-Making in Ship Design under Uncertainty
IMT-5-2020	Thomas H. Viuff	Uncertainty Assessment of Wave-and Current-induced Global Response of Floating Bridges
IMT-6-2020	Fredrik Mentzoni	Hydrodynamic Loads on Complex Structures in the Wave Zone
IMT-7-2020	Senthuran Ravinthrakumar	Numerical and Experimental Studies of Resonant Flow in Moonpools in Operational Conditions
IMT-8-2020	Stian Skaalvik Sandøy	Acoustic-based Probabilistic Localization and Mapping using Unmanned Underwater Vehicles for Aquaculture Operations
IMT-9-2020	Kun Xu	Design and Analysis of Mooring System for Semi-submersible Floating Wind Turbine in Shallow Water
IMT-10-2020	Jianxun Zhu	Cavity Flows and Wake Behind an Elliptic Cylinder Translating Above the Wall
IMT-11-2020	Sandra Hogenboom	Decision-making within Dynamic Positioning Operations in the Offshore Industry – A Human Factors based Approach
IMT-12-2020	Woongshik Nam	Structural Resistance of Ship and Offshore Structures Exposed to the Risk of Brittle Failure
IMT-13-2020	Svenn Are Tutturen Værnø	Transient Performance in Dynamic Positioning of Ships: Investigation of Residual Load Models and Control Methods for Effective Compensation
IMT-14-2020	Mohd Atif Siddiqui	Experimental and Numerical Hydrodynamic Analysis of a Damaged Ship in Waves
IMT-15-2020	John Marius Hegseth	Efficient Modelling and Design Optimization of Large Floating Wind Turbines

IMT-16-2020	Asle Natskår	Reliability-based Assessment of Marine Operations with Emphasis on Sea Transport on Barges
IMT-17-2020	Shi Deng	Experimental and Numerical Study of Hydrodynamic Responses of a Twin-Tube Submerged Floating Tunnel Considering Vortex-Induced Vibration
IMT-18-2020	Jone Torsvik	Dynamic Analysis in Design and Operation of Large Floating Offshore Wind Turbine Drivetrains
IMT-1-2021	Ali Ebrahimi	Handling Complexity to Improve Ship Design Competitiveness
IMT-2-2021	Davide Proserpio	Isogeometric Phase-Field Methods for Modeling Fracture in Shell Structures
IMT-3-2021	Cai Tian	Numerical Studies of Viscous Flow Around Step Cylinders
IMT-4-2021	Farid Khazaeli Moghadam	Vibration-based Condition Monitoring of Large Offshore Wind Turbines in a Digital Twin Perspective
IMT-5-2021	Shuashuai Wang	Design and Dynamic Analysis of a 10-MW Medium-Speed Drivetrain in Offshore Wind Turbines
IMT-6-2021	Sadi Tavakoli	Ship Propulsion Dynamics and Emissions
IMT-7-2021	Haoran Li	Nonlinear wave loads, and resulting global response statistics of a semi-submersible wind turbine platform with heave plates
IMT-8-2021	Einar Skiftestad Ueland	Load Control for Real-Time Hybrid Model Testing using Cable-Driven Parallel Robots
IMT-9-2021	Mengning Wu	Uncertainty of machine learning-based methods for wave forecast and its effect on installation of offshore wind turbines
IMT-10-2021	Xu Han	Onboard Tuning and Uncertainty Estimation of Vessel Seakeeping Model Parameters
IMT-01-2022	Ingunn Marie Holmen	Safety in Exposed Aquaculture Operations
IMT-02-2022	Prateek Gupta	Ship Performance Monitoring using In-service Measurements and Big Data Analysis Methods
IMT-03-2022	Sangwoo Kim	Non-linear time domain analysis of deepwater riser vortex-induced vibrations
IMT-04-2022	Jarle Vinje Kramer	Hydrodynamic Aspects of Sail-Assisted Merchant Vessels
IMT-05-2022	Øyvind Rabliås	Numerical and Experimental Studies of Maneuvering in Regular and Irregular Waves

IMT-06-2022	Pramod Ghimire	Simulation-Based Ship Hybrid Power System Conspet Studies and Performance Analyses
IMT-07-2022	Carlos Eduardo Silva de Souza	Structural modelling, coupled dynamics, and design of large floating wind turbines
IMT-08-2022	Lorenzo Balestra	Design of hybrid fuel cell & battery systems for maritime vessels
IMT-09-2022	Sharmin Sultana	Process safety and risk management using system perspectives – A contribution to the chemical process and petroleum industry
IMT-10-2022	Øystein Sture	Autonomous Exploration for Marine Minerals
IMT-11-2022	Tiantian Zhu	Information and Decision-making for Major Accident Prevention – A concept of information- based strategies for accident prevention
IMT-12-2022	Siamak Karimi	Shore-to-Ship Charging Systems for Battery- Electric Ships
IMT-01-2023	Huili Xu	Fish-inspired Propulsion Study: Numerical Hydrodynamics of Rigid/Flexible/Morphing Foils and Observations on Real Fish
IMT-02-2023	Chana Sinsabvarodom	Probabilistic Modelling of Ice-drift and Ice Loading on Fixed and Floating Offshore Structures
IMT-03-2023	Martin Skaldebo	Intelligent low-cost solutions for underwater intervention using computer vision and machine learning
IMT-04-2023	Hans Tobias Slette	Vessel operations in exposed aquaculture – Achieving safe and efficient operation of vessel fleets in fish farm systems experiencing challenging metocean conditions
IMT-05-2023	Ruochen Yang	Methods and models for analyzing and controlling the safety in operations of autonomous marine systems
IMT-06-2023	Tobias Rye Torben	Formal Approaches to Design and Verification of Safe Control Systems for Autonomous Vessels

