Rule-compliant and Fault-Tolerant Motion Planning
With Application to Autonomous Surface Vehicles

Tsolakis, A.

DOI
10.4233/uuid:6d3b427d-e1fc-488c-a2f8-9fa011d63520

Publication date
2025

Document Version
Final published version

Citation (APA)
Tsolakis, A. (2025). *Rule-compliant and Fault-Tolerant Motion Planning: With Application to Autonomous Surface Vehicles*. [Dissertation (TU Delft), Delft University of Technology].
https://doi.org/10.4233/uuid:6d3b427d-e1fc-488c-a2f8-9fa011d63520

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Rule-Compliant and Fault-Tolerant Motion Planning

With Application to Autonomous Surface Vehicles

Anastasios Tsolakis

Rule-compliant and Fault-Tolerant Motion Planning

With Application to Autonomous Surface Vehicles

Rule-compliant and Fault-Tolerant Motion Planning

With Application to Autonomous Surface Vehicles

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board of Doctorates,
to be defended publicly on
Tuesday 10 June 2025 at 10:00 o'clock

by

Anastasios TSOLAKIS

Master of Science in Systems & Control,
Delft University of Technology, The Netherlands,
born in Patras, Greece.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. R. R. Negenborn	Delft University of Technology, promotor
Dr. V. Reppa	Delft University of Technology, copromotor
Dr. L. Ferranti	Delft University of Technology, copromotor
Independent members:	
Prof. dr. B. Clement	École Nationale Supérieure de Techniques Avancées, France
Prof. dr. ir. P.H.A.J.M. van Gelder	Delft University of Technology, The Netherlands
Prof. dr. T. A. Johansen	Norwegian University of Science and Technology, Norway
Dr. F. L. Stevens	Erasmus University Rotterdam, The Netherlands
Prof. dr. M.T.J. Spaan	Delft University of Technology, The Netherlands, *reserve member*

Published and distributed by: Anastasios Tsolakis
Email: tas.tsolakis@gmail.com

Keywords: Trajectory optimization, traffic rules, autonomous surface vessels, fault diagnosis, fault-tolerant control, robust-adaptive model predictive control
Cover: Sketched with OpenAI's DALL·E
Printed by: Gildeprint

ISBN: 978-94-6384-793-3

An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`.

Contents

Summary

The development of Autonomous Vehicles (AVs) is revolutionizing multiple sectors, including automotive and maritime transportation. These innovations promise enhanced safety, operational efficiency, and environmental sustainability. To fulfill these promises, the navigation modules of these vehicles must be capable of handling complex environments, responding to unexpected events, and ensuring reliable decision-making under uncertainty. In this thesis, the focus is on Autonomous Surface Vessels (ASVs), aiming to enable safe and reliable navigation in complex and mixed-traffic environments (i.e., a mix of autonomous and non-autonomous vehicles) where uncertainties and faults pose significant challenges.

The central objective of this research is to develop fault-tolerant and rule-compliant motion planning algorithms for ASVs. These algorithms must be capable of ensuring safety even amidst component faults and other uncertainties while maintaining adherence to established traffic regulations. This capability is crucial to facilitate the coexistence of autonomous and human-operated vessels, especially during the transitional phase where human presence is predominant in maritime traffic.

The thesis is structured around the following key contributions:

1. **Model Predictive Trajectory Optimization for Rule Compliance**: The first step in ensuring safe navigation involves integrating maritime traffic rules into trajectory optimization. This thesis introduces a Model Predictive Contouring Control (MPCC) approach tailored to ASVs, which incorporates the International Regulations for Preventing Collisions at Sea (COLREGs). The proposed method utilizes affine constraints to formalize rule-compliant behavior within an optimization framework, allowing ASVs to navigate safely in dense and dynamic traffic scenarios. This rule-based approach enhances predictability and safety, enabling the ASV to interact seamlessly with human-operated vessels.

2. **Fault Diagnosis for Enhanced Operational Reliability**: The thesis addresses actuator faults that can compromise an ASV's performance. Operational reliability refers to the ASV's ability to function safely despite faults and uncertainties, ensuring continued safe navigation in dynamic environments. To achieve this, a model-based Fault Diagnosis (FD) method is developed, using residual analysis and adaptive thresholds to detect and isolate actuator faults in real-time. Simulation results demonstrate its effectiveness in distinguishing faults from regular disturbances and maintaining safety under faulty conditions.

3. **Set-Membership Estimation for Robust Fault Parameter Identification**: Expanding on the fault diagnosis capabilities, the thesis enhances the Set-Membership Estimation (SME) approach to robustly estimate fault parameters under the influence of uncertainties and noise. This method leverages the concept of unfalsified parameter sets, extending it to nonlinear systems affected by both state disturbances

and measurement inaccuracies. The estimation process is crucial for accurately identifying the extent of faults, enabling the system to adapt its behavior accordingly.

4. **Fault-Tolerant Trajectory Optimization and Control with Contingency Planning and Robust Adaptive Model Predictive Control**: Building upon the proposed fault diagnosis and parameter estimation methods, the thesis proposes a comprehensive motion planning framework that ensures both fault tolerance and rule compliance. The developed Robust Adaptive Model Predictive Control (RAMPC) integrates fault information into the trajectory optimization process, allowing the ASV to dynamically adjust its path in response to detected faults while continuing to adhere to traffic rules. This dual capability ensures that the ASV remains a reliable and predictable participant in mixed-traffic environments, even under faulty conditions.

The contributions presented in this thesis have been validated through extensive simulation studies, involving various traffic scenarios and fault conditions. The proposed algorithms have demonstrated the ability to navigate safely in congested environments, handle unexpected events, and adapt to faults without sacrificing safety or compliance with maritime regulations. The framework is implemented in ROS, with the controller developed in C++ and the ASV and other vessel simulators in Python. This choice streamlines future integration with real-world platforms and facilitates experimental validation. This research not only advances the state of the art in fault-tolerant motion planning for ASVs but also provides a solid foundation for broader autonomous navigation applications.

Samenvatting

De ontwikkeling van Autonomous Vehicles (AV's) brengt een revolutie teweeg in verschillende sectoren, waaronder de auto- en maritieme transportsector. Deze innovaties beloven verbeterde veiligheid, operationele efficiëntie en milieuduurzaamheid. Om deze beloftes waar te maken, moeten de navigatiemodules van deze voertuigen in staat zijn om complexe omgevingen aan te kunnen, onverwachte gebeurtenissen het hoofd te bieden en betrouwbare besluitvorming te waarborgen onder onzekerheid. In dit proefschrift ligt de focus op Autonomous Surface Vehicles (ASV's), met als doel veilige en betrouwbare navigatie mogelijk te maken in complexe omgevingen met gemengd verkeer (d.w.z. een mix van autonome en niet-autonome vaartuigen), waarin onzekerheden en fouten aanzienlijke uitdagingen vormen.

Het centrale doel van dit onderzoek is het ontwikkelen van fouttolerante en regelconforme bewegingsplanningsalgoritmen voor ASV's. Deze algoritmen moeten veiligheid garanderen, zelfs bij componentstoringen en andere onzekerheden, terwijl ze blijven voldoen aan de geldende verkeersregels. Deze capaciteit is van cruciaal belang om de coëxistentie van autonome en door mensen bestuurde vaartuigen mogelijk te maken, met name tijdens de overgangsperiode waarin menselijke aanwezigheid nog dominant is in het maritieme verkeer.

Het proefschrift is opgebouwd rond de volgende belangrijke contributies:

1. **Modelvoorspellende Trajectoptimalisatie voor Regelconformiteit**: De eerste stap richting veilige navigatie is het integreren van maritieme verkeersregels in de trajectoptimalisatie. Dit proefschrift introduceert een Model Predictive Contouring Control (MPCC) benadering, specifiek afgestemd op ASV's, die de International Regulations for Preventing Collisions at Sea (COLREGs) incorporeert. De voorgestelde methode maakt gebruik van affiene beperkingen om regelconform gedrag formeel vast te leggen binnen een optimalisatiekader, waardoor ASV's veilig kunnen navigeren in drukke en dynamische verkeersscenario's. Deze regelgebaseerde aanpak vergroot de voorspelbaarheid en veiligheid, en maakt naadloze interactie mogelijk met door mensen bestuurde vaartuigen.

2. **Foutdiagnose voor Verbeterde Operationele Betrouwbaarheid**: Dit proefschrift behandelt actuatorstoringen die de prestaties van een ASV kunnen ondermijnen. Operationele betrouwbaarheid verwijst naar het vermogen van een ASV om veilig te blijven functioneren ondanks storingen en onzekerheden, en zo veilige navigatie te waarborgen in dynamische omgevingen. Daartoe wordt een modelgebaseerde Fault Diagnosis (FD) methode ontwikkeld, waarbij gebruik wordt gemaakt van residuanalyse en adaptieve drempelwaarden om actuatorstoringen in real-time te detecteren en te isoleren. Simulatieresultaten tonen de effectiviteit aan van deze methode in het onderscheiden van storingen ten opzichte van gewone verstoringen, en het behouden van veiligheid onder foutieve omstandigheden.

3. **Set-Membership Schatting voor Robuuste Foutparameteridentificatie**: Ter uitbreiding van de foutdiagnosecapaciteiten, verbetert het proefschrift de Set-Membership Estimation (SME) methode om foutparameters robuust te schatten onder invloed van onzekerheden en ruis. Deze methode maakt gebruik van het concept van niet-weerlegde parameterverzamelingen en breidt dit uit naar niet-lineaire systemen die worden beïnvloed door zowel toestandsverstoringen als meetonnauwkeurigheden. Dit schattingsproces is essentieel voor het nauwkeurig vaststellen van de ernst van storingen, waardoor het systeem zijn gedrag dienovereenkomstig kan aanpassen.

4. **Fouttolerante Trajectoptimalisatie en -controle met Contingentieplanning en Robuuste Adaptieve Modelvoorspellende Controle**: Voortbouwend op de voorgestelde methoden voor foutdiagnose en parameterinschatting, stelt dit proefschrift een uitgebreid bewegingsplanningskader voor dat zowel fouttolerantie als regelconformiteit waarborgt. De ontwikkelde Robust Adaptive Model Predictive Control (RAMPC) integreert foutinformatie in het trajectoptimalisatieproces, waardoor de ASV zijn pad dynamisch kan aanpassen in reactie op gedetecteerde storingen, terwijl hij blijft voldoen aan de verkeersregels. Deze dubbele capaciteit zorgt ervoor dat de ASV een betrouwbare en voorspelbare deelnemer blijft in gemengde verkeersomgevingen, zelfs onder foutomstandigheden.

De contributies in dit proefschrift zijn gevalideerd via uitgebreide simulatiestudies, waarin diverse verkeersscenario's en foutcondities zijn onderzocht. De voorgestelde algoritmen hebben aangetoond veilig te kunnen navigeren in drukke omgevingen, onverwachte gebeurtenissen te kunnen verwerken en zich aan te kunnen passen aan storingen zonder concessies te doen aan veiligheid of naleving van maritieme regelgeving. Het raamwerk is geïmplementeerd in ROS, waarbij de controller in C++ is ontwikkeld en de ASV- en andere vaartuigsimulatoren in Python. Deze keuze vergemakkelijkt toekomstige integratie met fysieke systemen en experimentele validatie. Dit onderzoek levert niet alleen een contributie aan de stand van de techniek op het gebied van fouttolerante bewegingsplanning voor ASV's, maar vormt ook een solide basis voor bredere toepassingen binnen autonome navigatie.

Acknowledgments

As I write this section, my thoughts drift back to September 2018, the beginning of a journey that started with my move to Delft to pursue a Master's in Systems & Control. I was filled with excitement to be in this new place, away from home. Inspired by the academic environment, I extended my journey, this time embarking on a PhD in Cognitive Robotics. Now, six years later, as I complete my dissertation, those years play back like an old film—distant yet vivid. It's the people I've been fortunate to meet along the way who have turned this chapter of my life into a story I will always look back on with a smile.

First and foremost, I want to express my deepest gratitude to my daily supervisors, Dr. Laura Ferranti and Dr. Vasso Reppa, for their unwavering support over the years. They made this journey not only fruitful but genuinely enjoyable, always encouraging my ideas, providing critical feedback, and being there—countless times—by my side, understanding challenges and sharing their expertise. I also extend my sincere thanks to my promotor, Prof. Rudy R. Negenborn, for giving me the opportunity to start my PhD here and for his continuous guidance and wisdom throughout the journey. I am truly grateful to all three of them, not only for trusting me with this opportunity but also for embodying what it means to be kind, understanding, and compassionate mentors during my most challenging times.

I am also indebted to Prof. Benoit Clement, Prof. Pieter van Gelder, Prof. Tor Arne Johansen, and Dr. Frank Stevens, who served as my PhD committee members. Their valuable insights and feedback significantly enhanced the quality of this thesis, and I am grateful for the time and effort they invested.

The atmosphere here in Delft has been a constant source of motivation and inspiration. I'd like to thank the people I encountered throughout my time here in a way that mirrors the timeline of my journey. I still vividly remember those early days in 2018, meeting Dimitris, Tomasso, Elia, and Corrado. Working alongside them at DCSC was a joy, and I learned so much during our time together. My heartfelt thanks also go to my master's thesis supervisor, Prof. Tamas Kevicky, for his guidance in my initial research steps, and to Oscar, who co-supervised my thesis and played a pivotal role in my choice to pursue a PhD in Cognitive Robotics.

As my journey transitioned to CoR, I was fortunate to be surrounded by brilliant and kind-hearted peers. I am grateful to Hai and Bruno, who offered invaluable survival tips when I first arrived, fresh and slightly disoriented after being isolated for months due to the pandemic. To the rest of the "senior" generation—Oscar, Rodrigo, Giovanni, Max, Corrado, and Bas—I owe thanks for the enriching discussions and for always being approachable and supportive. I extend a special note of appreciation to my (time-varying) office mates—Italo, Lasse, Tomas, Andreu, Gang, Elia, Sihao, and Lorenzo—who made daily life at the office enjoyable. And, of course, to Jelle, my office mate from day one, whose humor brightened each day. My gratitude also goes out to Dennis, Max, Saray, Luzia, Sihao, Anna, Khaled, Mariano, Gustavo, Julian, Yujie, Ashwin, Kate, Julian, and Hidde for all the memorable moments both within and beyond the university. Our conversations—whether

over lunches, coffee breaks, or during our "Socratic Walks" on campus—spanned technical challenges, urgent societal issues, and even spirited debates on politics. To my colleagues from MTT—Nikos, Abishek, and Andrea—thank you for your insightful feedback during group meetings and for the wonderful experiences at conferences. The intellectual depth and kindness of everyone I met have been both humbling and inspiring. Thanks to all of you, the university has been a place I loved coming to each day, and I sincerely hope our paths will cross again.

Outside of the university, I was equally fortunate to be surrounded by good friends. I am deeply thankful to Giorgos and Vagelis for their warm welcome to the Netherlands and for supporting me through the challenging early days when I couldn't find a place to stay. They were my first anchors in this new place, and their help meant the world. My heartfelt thanks go to all my friends: Ntatsis, Agis, Apostolis, Achilleas, Vasia, Chris, Napo, and Stelios for everything they brought into my life here. The laughter and teasing, the bike trips, the shared adventures that became all the more essential during the pandemic, the music gatherings, and the comforting presence of a Greek community abroad. I am also very grateful to my dear Dutch friend, Rykiel, the best neighbor anyone could ask for, for the good times we shared. These are friendships I know will endure well beyond the end of this journey.

Finally, I would like to thank my family, Yannis, Xanthi, Chrysanthi, and Marianna, for their love and support throughout this entire journey. Without them, none of this would have been possible.

Tasos
Delft, June 2025

1

Introduction

1

1.1 Motivation & Challenges

Over the past few decades, we have witnessed our society moving rapidly towards an increased level of automation. Initially adopted in industry, automation solutions have aided in the design of more cost and energy-efficient means of production. While automation systems have traditionally been deployed in deterministic environments for repetitive tasks, recent advances in sensing and computing technology now enable their use in more complex and dynamic settings, extending their applications to various aspects of human activity.

The automotive industry is one of the most prominent areas to benefit from automation, with great research efforts both in academia and in industry. Among the main societal benefits presumed, the most interesting ones concern increased safety, improved traffic efficiency, and better mobility options for example with on-demand ride-sharing solutions (Figure 1.1). According to [1], road accidents are listed as one of the leading causes of fatalities, accounting for nearly 1.2 million deaths each year with an additional 20-50 million people suffering non-fatal injuries which often result in long-term disabilities [2]. An estimated 94% (±2%) of these accidents are attributed to human error [3] including impaired drivers due to alcohol consumption, speeding, distraction, and fatigue. Self-driving car technology has then the potential to increase road safety while at the same time improving traffic efficiency and mobility. Services such as on-demand ride-sharing can provide greater mobility options to people who are unable to drive due to disabilities or other reasons and also make commuting much more time and energy-efficient, especially in densely populated urban environments. Automotive transportation is undergoing a major transformation towards automation with a great societal impact in terms of economic and environmental sustainability.

Figure 1.1: Autonomous vehicle technology is set to revolutionize urban mobility, offering on-demand ride-sharing services that promise reduced emissions, greater accessibility, minimized urban parking space requirements, and enhanced safety for all users. [4]

Figure 1.2: Yara Birkeland is the world's first fully electric and autonomous container vessel with zero emissions. With this container vessel, Yara will reduce diesel-powered truck haulage by 40.000 journeys (approximately 1.000 tonnes of CO2) a year. [5]

While the automotive industry has had the leading role in this trend, the maritime sector is also progressing towards developing and utilizing autonomous maritime systems in many applications including transportation, large-scale monitoring, and search and rescue missions. This shift towards autonomy is motivated by numerous potential benefits such as greater efficiency, reduced operational costs, and increased safety. According to

[6], over the period 2014-2020, accidents of navigational nature (collisions, contacts, and groundings/strandings) represented almost 43% of all occurrences. Since human actions accounted for almost 61% of the contributing factors, autonomous maritime navigation may significantly reduce the risk of collisions which often lead to human casualties, damaged property, and devastating environmental disasters. In addition, many maritime tasks including freight transport (Figure 1.2), environmental monitoring (Figure 1.3), and hydrographic surveying (Figure 1.4), require the crew to be away from shore for extended periods and often under dangerous weather conditions. Autonomous solutions can significantly reduce risk exposure, time away from shore, and lack of family contact for extended periods which greatly influences personnel's social life.

Figure 1.3: The "Saildrone" unmanned surface vessel collects scientific-grade data from extreme ocean environments, aiding in climate research, weather forecasting, and ecosystem monitoring with zero operational carbon footprint. [7]

Figure 1.4: The "Otter" USV by Maritime Robotics is designed for hydrographic surveying and bathymetric mapping, offering a cost-effective, portable solution for precise data collection in sheltered and shallow waters. [8]

Moreover, it is often supported that autonomy in maritime vessels can lead to increased energy efficiency and reduced gas emissions mainly in two ways. First, autonomous ships can allow for increased slow-steaming[1] which in turn can lead to considerable reductions in fuel consumption. For example in some routes, a speed reduction of 5 knots can reduce fuel consumption by 54% [9]. Second, autonomy can lead to the ground-up redesign of ships since many personnel-related parts of a vessel (e.g., quarters, mess, stairs, bridge) are going to be redundant. This removes design constraints to a point that ships can have greater cargo capacity and less wind-resistant exterior design decreasing the specific CO_2 emissions per tonne-km significantly.

Last but not least, maritime automation can give rise to advanced cooperation frameworks which in turn can yield numerous benefits: Cooperation among autonomous vessels can further enhance safety by exploiting communication (sharing intentions and other safety-critical information) and also improve traffic efficiency by coordinating their voyage plans with infrastructure scheduling in order to avoid congestion at ports and make better use of infrastructure resources [10]. Therefore, the huge potential benefits of automation

[1]The term "slow-steaming" describes the deliberate reduction in the cruising speed of a sea vessel which is primarily done to reduce fuel consumption and pollution from emissions. Although lowering speed reduces the power requirements, the overall benefits of speed reduction may be limited by other factors, such as economically viable total voyage time.

1

in the maritime sector justify the recent growth in research efforts we have witnessed in the last decade.

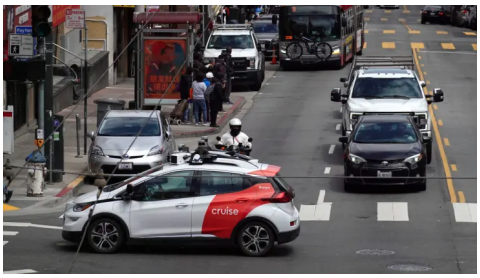

Figure 1.5: An autonomous car caused a significant traffic jam highlighting challenges in autonomous driving in urban traffic. [11]

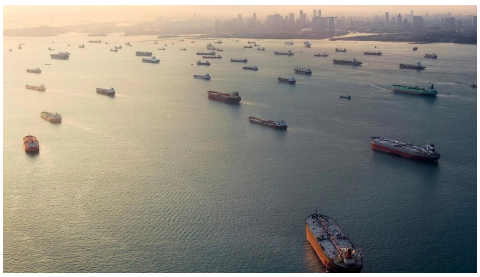

Figure 1.6: Marine traffic is also quite challenging, especially near ports and inland waterways where multiple traffic participants need to cooperate. [12]

Despite the numerous benefits that autonomy has to offer in both the automotive and maritime industries, it still remains a challenging task from a technological perspective. In both areas, there will be a transition period in which Autonomous Vehicles[1] (AVs) will be expected to *co-exist* with human-operated vehicles. This gives rise to major societal concerns about the capabilities of AVs to *interact safely* with non-autonomous vehicles (operated by humans) in *mixed-traffic* conditions and to handle *unexpected events*, such as *faults*, without causing disruptions or jeopardizing human safety. State-of-the-art solutions rely either on the full knowledge of the other autonomous vehicles' intentions using communication or on simplifying assumptions such as assuming constant velocity for the other traffic participants. The latter results in reactive collision avoidance actions where the autonomous vehicle avoids collision in a passive, unstructured manner without consideration of the traffic environment. The main difficulty in developing a safe motion planning algorithm in mixed-traffic scenarios is the *limited available information* among the vehicles that complicates decision-making and proactive planning.

As AVs will heavily rely on components such as sensors, actuators, computation units, and various other sub-systems, a major concern revolves around the potential consequences of component faults or complete failures during operation. Moreover, the existence of underlying uncertainties in the measurements or models used can severely deteriorate the performance or even pose considerable risks to the other traffic participants. For instance, in March 2018, a self-driving Uber vehicle involved in a fatal collision (Figure 1.7) in Tempe, Arizona, failed to properly identify a pedestrian crossing the street at night, partly due to issues with the vehicle's perception system which includes sensors and cameras [13]. Additionally, Tesla faced scrutiny when more than 40.000 vehicles were recalled due to a software update that miscalibrated the power steering, potentially leading to loss of steering assist and increased risk of accidents [14]. This recall was highlighted

[1]The term "Autonomous Vehicles" is used in the general context in this report to describe both automotive vehicles and maritime vessels that is, vehicles that navigate in dense traffic environments shared with other traffic participants (Figures 1.5 and 1.6). When using the word "autonomous" we refer to at least level-4 autonomy according to SAE and IMO that is, we assume *full autonomy*: The operating system is able to make decisions and determine actions by itself in all conditions.

by unexpected steering commands that could force drivers to exert much more effort to control the vehicle, especially at lower speeds. The maritime industry has witnessed similar issues for conventional vessels, such as in 2017 when a navy ship collided with a container ship off the coast of Japan, a mishap partly attributed to failures in its navigation system's radar sensors [15]. More recently, in March 2023, a container ship experienced a catastrophic power failure that led to a collision with the Francis Scott Key Bridge in Baltimore (Figure 1.8), resulting in human casualties and significant disruptions [16]. These incidents underscore the critical need for robust fault diagnosis and fault-tolerant systems to ensure the safety and reliability of autonomous technologies.

Figure 1.7: "In 2019, a fault in a self-driving car led to a deadly crash, emphasizing the need for improved safety measures in autonomous driving technology. [17]

Figure 1.8: In 2024, a power failure on a container ship led to a complete loss of maneuverability, resulting in a crash that caused the Baltimore bridge to collapse. [18]

In conclusion, while AVs have the potential to significantly improve transportation in many aspects, the aforementioned challenges underscore the need for advancements in AV technology to ensure reliable operation under varied and unpredictable conditions. The research goal of the thesis is *to develop efficient, online motion planning algorithms to allow autonomous vehicles to safely navigate in mixed traffic, i.e., among human-operated vehicles even in the presence of faults.* Our algorithms are initially developed for Autonomous Surface Vessels (ASVs) as the primary platform, but they are designed to be adaptable for broader applications in autonomous navigation.

1.2 Research Questions

The main research question of this thesis is:

How can ASVs safely navigate in mixed traffic environments even in the presence of faults?

To answer the question above, we need to answer the following sub-questions:

Q1: How can ASVs navigate safely and efficiently in dense traffic environments while ensuring compliance with maritime traffic rules?

Q2: How to detect and isolate actuator faults in ASVs to enhance overall operational safety and reliability?

1

Q3: How can fault parameters be accurately and robustly estimated under varying operational conditions, including the presence of disturbances and noise?

Q4: How can we jointly guarantee fault-tolerant and rule-compliant trajectories for ASVs operating in mixed-traffic environments?

1.3 Approach

This thesis addresses the problem of fault-tolerant autonomous navigation within the context of "mixed-traffic" environments, characterized by the interaction of autonomous and human-operated vehicles. We assume that the state of the ASV including its pose and velocity, is accurately determined through a set of suitable sensors with sufficient precision. We further assume that another set of sensors effectively captures the environment of the ASV, such as the positions, velocities, and dimensions of both static and dynamic obstacles (i.e., other traffic participants). Further, we study the problem of faults in these environments and their unexpected occurrence during navigation. Specifically, we devise strategies to diagnose these faults and mitigate their effects.

The first goal is to devise a *local motion planning algorithm* that enables the ASV to navigate safely and efficiently, adhering to traffic regulations while avoiding collisions. This is particularly challenging due to the unpredictable nature of human-operated vehicles and the lack of direct communication about their intentions. To address this, we integrate traffic rules into the motion planning algorithm, using them as a proxy for communication. This allows traffic participants to infer others' likely actions based on simple observable metrics like pose and velocity, thus creating a framework of mutual expectations and obligations.

Unexpected events, such as system faults, can further challenge navigation and severely impact the ASV's operational capabilities. Therefore, the second goal of our approach involves enhancing the robustness of the motion planner to accommodate such faults without compromising safety or causing disruptions. We achieve this by developing a fault-tolerant planning strategy that ensures the AV remains a safe and compliant traffic participant under faulty conditions.

The methodologies developed in this thesis primarily rely on two foundational pillars designed to address the dual challenges of rule compliance and fault tolerance in motion planning.

1.3.1 Model Predictive Control

The backbone of the methods presented in this thesis is Model Predictive Control (MPC), also known as Receding Horizon Control (RHC) [19, 20]. MPC is fundamentally an approximation of an infinite horizon Optimal Control Problem (OCP). The central concept involves controlling a dynamical system by minimizing a cost function, which typically encapsulates the system's objectives while adhering to both state and input constraints. These constraints may arise from the physical properties of the system or be user-defined to encourage desired behaviors. Within a defined time horizon, the system's future states are predicted using an established dynamical model and, unlike traditional OCP, MPC tackles the optimization problem in real time over this finite horizon. At each control cycle, an "optimal" control sequence is calculated. Feedback is introduced by applying the first input from this sequence to the system, and the cycle repeats at the next iteration.

Utilizing MPC as a strategy for motion planning offers significant advantages. Firstly, it allows for proactive adjustments to changes in the system and its environment, transitioning from a reactive to a proactive control approach by accommodating predictions about the system and its environment. Additionally, OCP seamlessly integrates a wide range of constraints—from model dynamics and input saturation to state restrictions—directly into the optimization problem, eliminating the need for complex, cascaded control architectures. Lastly, MPC's framework readily extends to robust settings, effectively managing state and output uncertainties, and to adaptive settings that can address unexpected, time-varying effects such as faults.

1.3.2 Fault-Tolerant Control

In control systems, faults typically refer to the malfunction or improper operation of components such as sensors, actuators, and processes. Fault-Tolerant Control (FTC) is essential for detecting these faults and mitigating their effects to ensure continued operation or, at a minimum, controlled degradation of system performance. This capability is particularly critical in safety-sensitive applications, such as navigation among human-operated vehicles. FTC typically consists of two key components: a Fault Diagnosis (FD) module that detects, isolates and estimates faults, and a reconfiguration strategy that adjusts the controller based on FD information.

Both FD and FTC can be classified as passive or active. Passive FD relies solely on naturally occurring system inputs and outputs without modifying control actions, making it suitable for applications where external intervention is not feasible. In contrast, active FD enhances fault detectability by injecting test signals or modifying control inputs to provoke measurable fault effects. This approach is particularly useful in safety-critical applications where distinguishing between similar faults is essential. Similarly, FTC strategies can be categorized as passive or active. Passive FTC designs controllers to be inherently robust against faults, allowing the system to maintain stability without requiring reconfiguration. Active FTC, on the other hand, relies on an FD module to detect, isolate, and estimate faults, enabling adaptive responses to mitigate their effects dynamically, making it a less conservative approach.

In this work, we leverage the flexibility of MPC to implement active FTC by reconfiguring the controller in response to detected faults. This adaptation is supported by a robust active FD module capable of detecting, isolating, and estimating faults despite environmental disturbances, measurement noise, and model mismatches. Specific FD modules are developed to continuously monitor the system's operational conditions using input-output measurements and identify deviations from expected behavior. Building on this capability, the nominal MPC framework is extended to incorporate real-time system health data, enabling it to mitigate the impact of faults immediately after their detection and estimation.

1.4 Contributions

To reach the overall research goal established in Section 1.1 and answer the research questions of 1.1, this thesis presents the following scientific contributions:

1. A Model Predictive Trajectory Optimization and Control Algorithm Considering

Traffic Rules [21], as a response to Research Question **Q1**. The algorithm is tailored for the application of Autonomous Surface Vessels (ASVs), and it contains the following contributions concerning the state of the art on this topic:

- A formal derivation of affine constraints that guarantees rule compliance in a convex search space.
- Simplified transition expressions in the traffic rule decision-making module that rely on the design of the affine constraints.
- An algorithm that scales to multiple obstacles and allows the vessels to safely navigate through dense traffic environments.

2. An active FD method that can robustly detect and isolate actuator faults based on input-output measurements [22], as a response to Research Question **Q2**. The method relies on residuals generated by a nonlinear observer, coupled with adaptive thresholds and an active reconfiguration strategy of the MPC controller to enhance isolability. The contributions with respect to the state of the art are:

   - A planning-integrated active FD algorithm capable of detecting and isolating actuator faults, enhancing overall safety by proactively accounting for actuator faults.
   - Adaptive thresholds that adjust dynamically to the system's nonlinear behavior for more accurate fault detection, considering noise and disturbances.
   - Enhanced fault isolation using control redundancy and detailed model dynamics, reducing the dependency on additional control allocation modules.

3. A passive FD method to detect and estimate fault parameters along with their feasible set based on Set-Membership Estimation (SME) [23], as a response to Research Question **Q3**. Fault detection relies on inverse tests and the estimation of faulty parameters via the computation of their feasible parameter set. Key contributions include:

   - Set-membership estimation to nonlinear systems, accounting for both disturbances and measurement noise. This capability ensures *false alarm immunity* by design, thereby increasing the robustness of the fault detection process.
   - A tighter outer approximation of the feasible parameter set that balances accuracy and computational efficiency, based on user-defined preferences. This leads to improved *fault detectability*, reducing the risk of missed detections and enhancing the system's responsiveness to faults.
   - Adaptive regularization in fault parameter estimation to handle cases of sparse, non-informative measurement data, resulting in improved *fault identifiability*.

4. A fault-tolerant trajectory optimization and control framework that relies on a dual-plan strategy comprising a primary and contingency trajectory. Both plans are designed in a Robust Adaptive MPC (RAMPC) fashion that utilizes the FD of the previous Chapter and is able to guarantee rule-compliant, fault-tolerant trajectories for the ASV in mixed-traffic environments, as a response to Research Question **Q4**.

1.5 Outline

The outline of this thesis is shown in Figure 1.9. Chapter 2 briefly reviews the state of the art in rule-compliant motion planning and fault-tolerant motion planning. Chapter 3 presents a trajectory optimization algorithm for rule-compliant collision avoidance in dynamic, mixed-traffic environments (as response to Research Question **Q1**). Chapter 4 presents an FD method developed based on residuals and adaptive thresholds (as response to Research Question **Q2**) and Chapter 5 presents an FD method based on set membership estimation (as response to Research Question **Q3**). Chapter 6 combines proposes a novel rule-compliant and fault-tolerant motion planner in a RAMPC fashion (as response to Research Question **Q4**). Finally, Chapter 7 concludes the thesis and provides recommendations for future research.

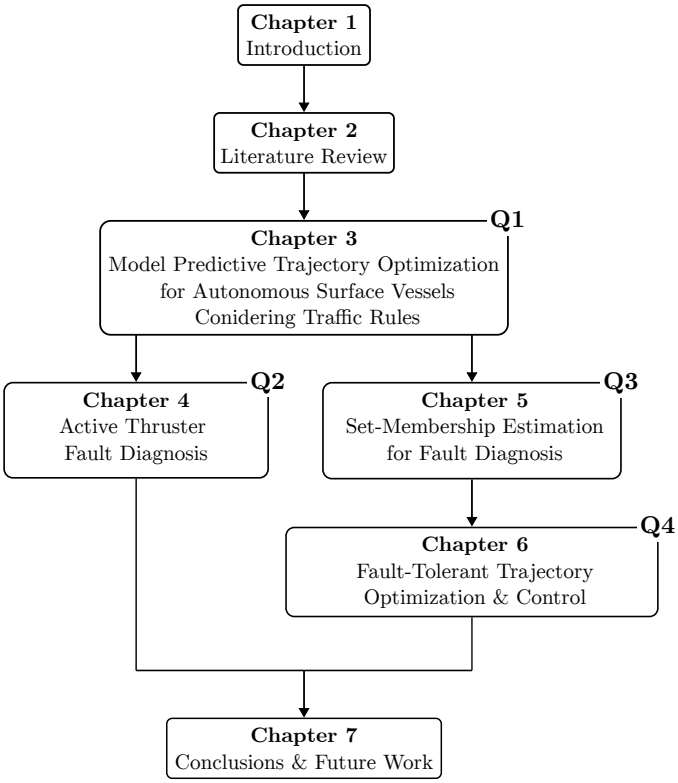

Figure 1.9: The outline of this thesis.

2

Literature Review

This thesis addresses fault-tolerant motion planning in mixed-traffic environments. To facilitate a deeper understanding of the subject, a comprehensive literature review is conducted to summarize the current state of the art and identify existing research gaps. The chapter is organized as follows: Section 2.1 provides a brief introduction to the topic of motion planning and its evolution as a field in robotics. Section 2.2 delves into the challenges of motion planning in mixed-traffic environments, particularly focusing on how traffic rules have been applied in maritime navigation to mitigate uncertainties related to traffic participants. Section 2.3 explores the issue of fault-tolerant control in motion planning, with a special emphasis on marine systems. Specifically, Section 2.3.1 examines the problem of fault diagnosis, while Section 2.3.2 reviews various techniques employed for fault accommodation that work in tandem with the diagnosis methods discussed. Fault diagnosis and fault tolerance methods in the scope of marine systems are discussed in Section 2.3.3. Finally, Section 2.4 summarizes the chapter and highlights the identified research gaps.

2.1 Motion Planning for Autonomous Vehicles

Operating mobile robots requires interdisciplinary expertise, combining knowledge from various fields to enable autonomous navigation in real-world environments. The main question it addresses is "How can a mobile robot move unsupervised through real-world environments to fulfill its task?" [24]. This question involves the solution and integration of many different sub-problems such as *perception*, *localization & mapping*, *cognition & path planning* and *motion control* which are often interconnected as illustrated in Figure 2.1.

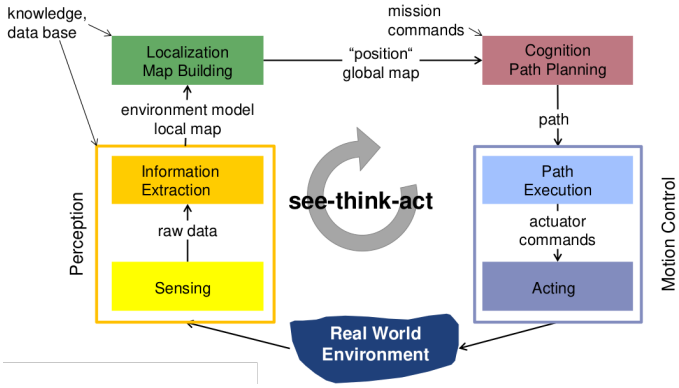

Figure 2.1: The "see-think-act" control scheme of a mobile robot. [24]

This work focuses specifically on cognitive path planning (i.e., high-level decision-making for trajectory generation and navigation in dynamic environments) (red block in Figure 2.1) and motion control (blue block in Figure 2.1), assuming there is sufficient information that we can exploit from the modules of perception and localization (yellow and green blocks, respectively, in Figure 2.1). Path planning for mobile robots is a well-established field in which various methodologies have been developed, spurred by both theoretical interest and numerous applications. As discussed in [24], traditionally there are two main competencies in mobile robot navigation: The first one is *Path Planning* which involves identifying a trajectory that will cause the robot to reach a desired goal position, given a map and its current position. The second one is *Obstacle Avoidance* which depends on real-time sensor readings to modulate the trajectory of the robot in order to avoid collisions. However, the complexity of motion planning in *Urban and Marine Environments* has led to algorithms in which these attributes are inter-wined and often combined with other important layers such as behavioral decision-making (usually included as part of the cognition task shown in the red block of Figure 2.1). Since the scope of existing motion planning algorithms is vast and varies a lot depending on the intended application, in the following sections we focus on motion planning algorithms specifically for urban and marine environments, in the context of mixed traffic, uncertainties, and faults.

According to [25], in motion planning for autonomous vehicles, the tasks of cognition, path planning, and motion control that are illustrated in Figure 2.1 are typically hierarchically structured into route planning, behavioral decision-making, local motion planning, and feedback control as shown in Figure 2.2. This modular representation that

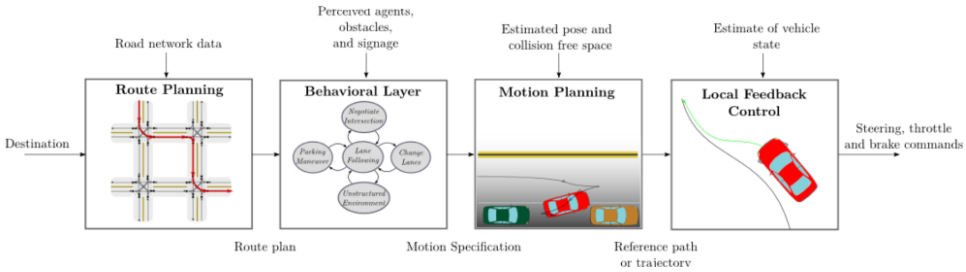

Figure 2.2: An hierarchical illustration (from left to right) of an AV's decision-making processes. [25]

spans from the high-level task of route planning to the lower-level task of local feedback control is especially helpful in discussing the suitability of various existing motion planning algorithms.

Given the current position and a desired destination, *route planning* is the task of selecting a route through a given map that describes the traffic network and is usually considered of a more "global" nature. This task is usually intended to be a one-time, pre-processing step that returns an "optimal" higher-level path to be followed by the autonomous vehicle [25]. Optimality criteria usually include a combination of traversed distance, time duration, and energy consumption. This task is usually realized by representing the road network as a directed graph with edge weights corresponding to the aforementioned criteria and then utilizing a graph search algorithm to compute the minimum cost path. Algorithms used for roadmap construction as a graph include Cell Decompositions [26], Visibility Graphs [27, 28] or RRG [29]. Classical graph search algorithms include Dijkstra [30], A* [31] and D* [32]. Lastly, in case discretization of the environment and consequent search is computationally forbidden, sampling-based methods such as PRM [33], RRT [34] and their variants are often employed.

Assuming that the autonomous vehicle is given the desired route in a global static map, it then needs to traverse this route in the traffic network while interacting with other traffic participants according to the prevailing conventions and traffic regulations. The *behavioral layer* is responsible for selecting the appropriate set of actions according to these regulations and the surrounding traffic participants. This module is especially difficult to design for two main reasons. First, since the traffic rules are intended for human operators, they are qualitative and too abstract for machine implementation in an algorithmic setting. A common approach to automate this decision-making module is via Finite State Machines (FSMs) (such as Moore or Mealy machines) by modeling each behavior as a state that depends on the traffic situation of the vehicle. Such approaches have been utilized in [35]. While this addresses the problem at first glance, it is questionable to what extent it is adequate as it is heavily based on heuristics. The second reason that makes the design of such a behavioral layer difficult is the fact that urban and marine traffic environments are characterized by an increased level of uncertainty over the intentions of the other traffic participants. This problem has been addressed with both model-based approaches relying for example on chance constraints [36, 37], MDPs [38, 39, 40, 41, 42] or game theory [43, 44, 45] and machine learning methods [46, 47, 48, 49]. Nevertheless, this module still remains one of the most challenging to design with respect to the complexity of the task and the

need of formal safety guarantees that are not usually addressed in existing works.

The *motion planning* module is usually responsible for computing a local path for the vehicle that follows the given, high-level route and also complies with the expected behavior and existing motion constraints. The main task in local motion planning is usually to provide a collision-free path with respect to both static (possibly uncharted obstacles) and dynamic obstacles that are inferred from real-time sensor readings. A taxonomy of obstacle avoidance techniques is described in [50]. In brief, the most notable mentions include *methods of physical analogies* which assimilate the obstacle avoidance problem, and *methods of subsets of controls* which compute an intermediate set of motion controls, and next they choose one of them as a solution. Among the methods of physical analogies, the APF method [51] and its extensions [52, 53, 54], are known for their widespread use along with some variants for example NFs [55, 56, 43]. The methods of subsets of controls can be further distinguished in two types: Methods that compute a subset of motion directions (VFH [57], ORM [58]) and methods that compute a subset of velocity controls like DWA [59] or VO [60] and its extensions (GVO [61], RVO [62], AVO [63], ORCA [64]). Last but not least, recent advances in computation technologies have rendered predictive control algorithms a favorable option for motion planning including both gradient-based methods such as MPCC [65, 66, 67], but also sampling-based methods such as MPPI [68, 69, 70].

The last module of *local feedback control* is responsible for the low-level control of the actuators of the vehicle so that it follows the desired, local, collision-avoiding path. This module is traditionally relying on classical control theory and the methods used span from those designed for linear (or linearized) systems (e.g., state feedback, PID, LQR, etc.) to those designed for nonlinear systems (e.g., Backstepping, SMC, IDA-PBC, etc.). Since low-level control often needs to optimize performance criteria in the presence of actuator limitations, predictive control (MPC, NMPC) is also commonly used if the system dynamics are adequately slow compared to the required control rate.

Although the decision-making scheme described in [25] is intended for autonomous urban vehicles, a similar approach is directly applicable to marine vessels. This is because marine vessels need route planning, behavioral layers, motion planning, and motion control as well. The research field of autonomous maritime navigation has adopted another control scheme for ASVs. This control scheme is known as the Guidance, Navigation, and Control (GNC) system, illustrated in Figure 2.3 for marine vessels. Although the control scheme of Figure 2.3 seems different than the one shown in Figure 2.2, the underlying concepts are essentially the same. The control block in Figure 2.3 corresponds to the local feedback control of Figure 2.2, the navigation block in Figure 2.3 acts in a similar manner as the behavioral layer block of Figure 2.2 since its task is situational awareness and lastly, the blocks of route and motion planning of Figure 2.2 are combined in the guidance block of Figure 2.3.

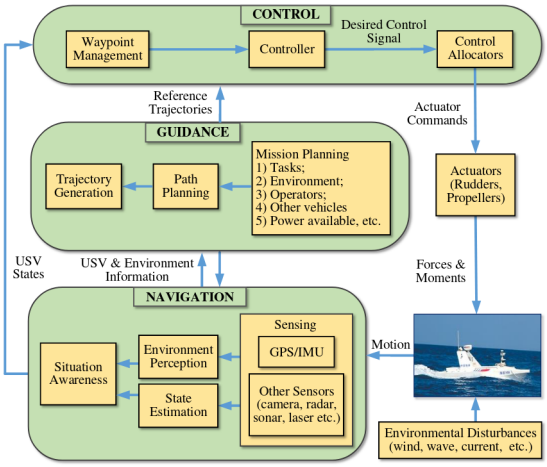

Figure 2.3: The general structure of an ASV's Guidance Navigation and Control System. [71]

2.2 Rule-Compliant Motion Planning

Uncertainties in the context of motion planning have various sources. The main ones include *i*) uncertainties in the motion of other traffic participants, *ii*) uncertainties that stem from sensor noise, *iii*) model uncertainties for model-based approaches and *iv*) disturbances that usually stem from exogenous factors (e.g., wind). Perhaps the largest source of uncertainty in mixed-traffic environments lies in the motion prediction of other traffic participants. Although there are various ways to improve these predictions as discussed in the previous section, there will still be uncertainties on the exact pose and intentions of the other participants. To account for such uncertainties, [46, 66] follow a probabilistic approach assuming a known posterior distribution describing the current and future state of the other vehicles up to a certain number of time steps in the future. Uncertain behaviors are formulated as a POMDP in [39, 41, 42] where they predict the probabilistic motion states of the other vehicles over a finite horizon. In [72] the uncertainties on the surrounding environment are bounded with a safety corridor that is generated from vehicle data gathered from a simulator. A reachability-analysis approach is found in [73] where they compute the reachable set of other traffic participants by assuming a known motion model. Illegal actions are then removed from the reachable set and an occupancy area that encloses the positions of the surrounding traffic participants is predicted.

2.2.1 Rule-compliance in Autonomous Vehicles

Inference of the other traffic participants' intentions is one of the main challenges in autonomous navigation. A structured environment can mitigate this difficulty as it describes expected actions explicitly. With this prospect, one of the main priorities for motion planning in mixed-traffic environments should be the incorporation of existing traffic rules. This is not only necessary for a real-world deployment but can further be exploited as implicit communication among the traffic participants. Incorporating a set of rules in a

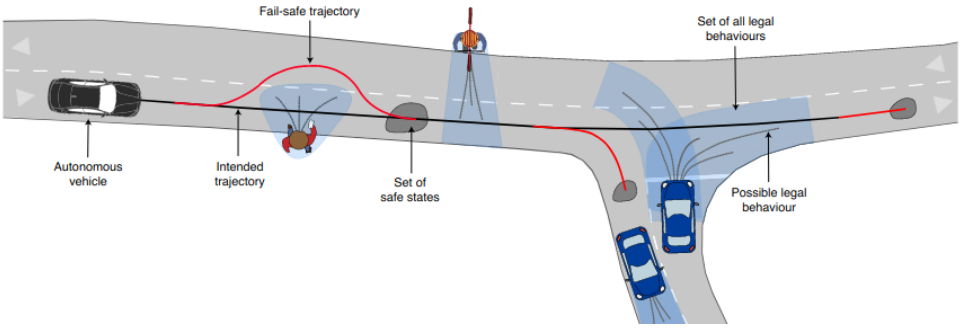

Figure 2.4: The occupancy areas are depicted as the blue areas that contain the most likely behaviors of other traffic participants (grey lines).[73]

motion planning algorithm is not straightforward however as it needs to merge "logical" statements (intended for human interpretation) with the technicalities of the corresponding algorithm. Consideration of traffic rules in mixed traffic environments is a topic that has drawn a lot of attention in the last few years. In [74] a formalization of the traffic rules is attempted via higher-order logic to ensure accountability of the autonomous vehicles. In [75, 76], reachability analysis in conjunction with formalized traffic rules is used to verify the safety of maneuvers of AVs while in [77] the authors develop an ethical trajectory planning algorithm with a framework that aims at a fair distribution of risk among road users while accounting for violations of traffic rules in the risk distribution. Formal guarantees for AVs are provided in [78] based on a behavioral contract that captures a set of explicitly defined assumptions about how all agents in the environment make decisions.

Considering traffic rules in mixed-traffic environments is not only indispensable for AVs so that they do not impede traffic, but can also be of great importance in the prediction of other traffic participants' intentions. Human drivers often *predict* the intentions of other drivers by exploiting existing knowledge of *traffic rules* which constructs implicit communication. In [79] the authors combine LTL with MDPs in a receding horizon fashion to to predict the intentions of the other vehicles and then plan their own actions with RRT*. The approach is tested in a simple road segment. In [80] they use CNNs for motion prediction by considering "domain knowledge" which essentially includes motion constraints and "rules of the road". Both constraints and rules place strong priors on likely motions. While motion prediction is fundamentally challenging due to the uncertainty over the intentions of other participants, their behaviors are constrained by both infrastructure (like road limits or traffic lights) and the actions of others (e.g., other vehicles crossing).

2.2.2 Rule-compliance in Autonomous Surface Vessels

The exploration of integrating traffic rules into motion planning algorithms extends beyond just autonomous vehicles on roads; it is equally pertinent to autonomous marine surface vessels. Safety in autonomous maritime navigation is a broad and active topic (refer to [81] for an overview). Ongoing research regarding safety has focused on the problem of interpretation and incorporation of the International Regulations for Preventing Collisions at Sea (COLREGs) [82] in autonomous navigation. Fuzzy logic [83], Dynamic Bayesian

Networks (DBN) [84, 85] as well as Finite-State Machines (FSM) [86] have been proposed for situational awareness and decision making. For the task of collision avoidance, methods of subsets of controls, such Velocity Obstacles (VO) [87, 88] and some extensions like Generalized Velocity Obstacles (GVO) [42], Probabilistic Velocity Obstacles (PVO) [89], Dynamic Reciprocal Velocity Obstacles (DRVO) [90] or Optimal Reciprocal Collision Avoidance (ORCA) [91] as well as methods of physical analogies, such as Artificial Potential Fields (APF) [92, 93, 94] have been studied thoroughly to work along with COLREGs, as they are methods of low computational complexity. This simplicity, however, comes at the cost of being more reactive and difficult to combine with the full set of traffic regulations which may require longer planning horizons. Moreover, these methods usually give a rough direction of where the ASV should move while disregarding vessel dynamics unless additional reachability approximations are used [90].

To plan over longer horizons, search-based methods like A* [95, 96, 97], Voronoi Diagrams [98], and optimal Rapidly-exploring Random Trees (RRT*) [99, 100] have also been employed. They search for a dynamically feasible path in a joint time-state space by either creating artificial costs or obstacles in the discrete grid map to resemble rule-compliant maneuvers. Because the trajectories are computed in the configuration space, they are often non-smooth and their computation is expensive. Moreover, these methods are hard to combine with the complete set of traffic regulations and may even ignore some of the rules in multi-vessel situations [99]. Recently, learning-based methods have also been investigated in conjunction with the traffic rules [101, 102], though drawbacks in these methods often include poor generalizability, convergence to local minima, and lack of formal guarantees.

A popular category for motion planning under COLREGs includes optimization-based methods. The main benefit of these methods is the potential to combine multiple objectives and constraints of different nature in a single control module. Among the limitations, the most important ones include deadlocks (due to the local nature of the computed path) and high computational demands (depending on the complexity of the formulated problem). To circumvent these limitations, [103] established a sample-based MPC approach that considers a finite space of control inputs. Unlike typical MPC formulations, these methods do not identify the best action at every time step during trajectory generation. This work was tested with extensive field verification [104, 105] and it was further extended in other research directions such as Scenario-Based MPC [106, 107]. In [108], the task of navigation under COLREGs is expressed as a multi-objective optimization problem where a particle swarm optimization algorithm is used for its solution. While these methods are suitable in cases of limited computation capacity, they are not guaranteed to converge, and thus, a collision-free path may not be found.

Optimization-based algorithms that rely on conventional gradient-based methods have been studied as well [109, 110, 111, 112] having the benefit of exploring the entire control input space. However, all aforementioned approaches rely on a heuristic cost function for rule compliance (that either combines hazard metrics or creates repulsive fields based on the geometrical situation). The use of soft constraints for safety-critical tasks such as rule compliance is questionable since there can be conflicts with other mission objectives (e.g., trajectory tracking). Works in which rule compliance is enforced by introducing hard state constraints to the optimization problem include [113, 114]. In [113] however, the

designed constraint is too conservative as it restricts the heading of the ASV and it does not take advantage of state predictions. In [114], hard constraints based on a half-space definition for the domain of the encountered vessels are defined based on their relative position with respect to the ASV and a deflection angle as a parameter. However, that particular definition can lead to infeasibilities since the position of the ASV is not taken into account while tuning this parameter. Moreover, the resulting constraints are nonlinear which may complicate the solution of the optimization problem.

Overall, while the topic of rule-compliant motion planning has spurred significant research activity, a critical gap remains in developing a straightforward implementation that fully integrates traffic rules into motion planning. Such an implementation should ensure rule compliance, generate dynamically feasible trajectories, and efficiently scale to multiple traffic participants, each considered as a dynamic obstacle. Current approaches, whether optimization-based, search-based, or learning-based, often fall short in at least one of these aspects, either due to computational complexity, lack of scalability, or insufficient integration of traffic rules. Addressing these challenges is essential for advancing the real-world deployment of autonomous navigation systems in both terrestrial and maritime environments.

2.3 Fault-Tolerant Motion Planning

Unexpected events such as *faults* differ from the aforementioned types of uncertainties in that they manifest at discrete points in time, rather than continuously, and can severely compromise the safe operation of a system. As autonomous systems expand into more sectors, they increasingly depend on sophisticated technology and complex hardware, escalating the intricacies of their operational framework. Relying on critical components like sensors, actuators, and computational units introduces significant safety and reliability challenges. Faults in these components can lead to failures, posing catastrophic risks and jeopardizing safety. Ensuring safety and reliability in these technologies is of paramount importance, necessitating robust mechanisms to manage and mitigate faults effectively. For these reasons, faults in robotic systems have long been a critical concern across various domains including robotic manipulators [115, 116, 117], ground [118, 119], marine [120, 121, 122, 22], aerial [123, 124, 125], and multi-agent systems [126]. Faults primarily undermine system performance because they can impair the controllability and observability of the system but also create a discrepancy between the system's theoretical model and the system itself. This mismatch compromises controller performance, potentially leading to hazardous behavior.

When referring to faults, the two main topics of interest are Fault Diagnosis (FD) and Fault Tolerance (FT), which are often interdependent and complement each other. FD is a diagnostic procedure involving the detection, isolation, and identification of faults, typically relying on monitoring the system's behavior. Fault diagnosis is crucial for understanding the health of a system and serves as a precursor to implementing corrective actions. FT, on the other hand, focuses on maintaining the control system's performance and stability despite the presence of faults. When the system adapts to faults using real-time information provided by an FD module, this is known as Active Fault Tolerance (AFT). This typically involves reconfiguring the control strategy or switching to a backup system. A schematic representation can be seen in Figure 2.5. The following sections will provide a detailed

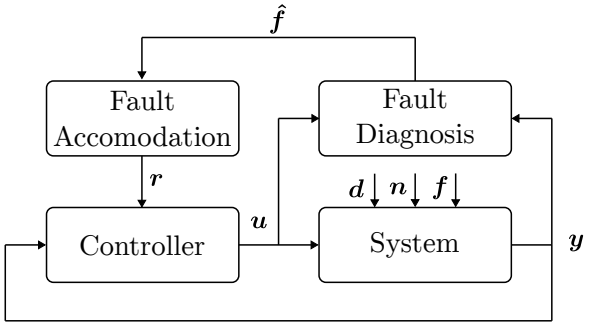

Figure 2.5: A schematic representation of the diagnosis and tolerance modules in an AFT control system. The FD module relies on input $\boldsymbol{u}$ and output measurements $\boldsymbol{y}$ to monitor the system's health while subjected to disturbances $\boldsymbol{d}$, noise $\boldsymbol{n}$ and faults $\boldsymbol{f}$. The FD module is responsible for detecting and diagnosing an estimate of the fault denoted as $\hat{\boldsymbol{f}}$ and updating the accommodation module with that information. The latter is then responsible for reconfiguring the controller with a reconfiguration signal $\boldsymbol{s}$ in order to adapt to the faulty conditions.

discussion on FD and FT, setting the foundation for identifying the most suitable methods to be used in conjunction with motion planning in the context of this thesis.

2.3.1 Fault Diagnosis in Robotics

FD focuses on monitoring a system's healthy operation and it is arguably the most challenging aspect regarding faults. The main challenge arises from various sources of discrepancies such as model mismatch, environmental disturbances, and measurement noise, which complicate the accurate detection, isolation, and identification of faults. To address these challenges, FD methods can be broadly categorized into several groups. Model-based methods leverage mathematical models of the system to generate and evaluate residuals or estimate parameters. Signal-based methods analyze the characteristics of system signals in various domains. Knowledge-based methods incorporate expert knowledge or data-driven approaches like machine learning. Hybrid methods combine different techniques to enhance diagnostic capabilities. Finally, process history-based methods utilize historical data and process trends to identify faults. Each category offers unique advantages and is suited to different types of systems and fault scenarios. As the autonomy of robots increases, there is an increasing importance of health monitoring that can be carried out quickly, online, and onboard based on limited computational power. A thorough review of FD for robotic systems can be found in [127] with a taxonomy of the main methods illustrated in Figure 2.6. In this study, it is highlighted that model-based methods are usually of preference as they usually pose a low computational burden compared to statistical data-driven approaches such as outlier detection, but the quality of diagnosis depends heavily on the fidelity of the model. Learning-based methods can offer quick online solutions if learning occurs offline but this produces static models which may not fit new behaviors. Online learning offers a dynamic model for FD but increases the computational load considerably.

Set-based methods have been increasingly popular for FD in robotics, mainly because they eliminate the need for knowing statistical distributions of unknown signals by relying solely on boundedness assumptions. In [128, 129] a bank of observers generates residual zonotopes and tests their inclusion inside corresponding invariant sets. In [130] a new class

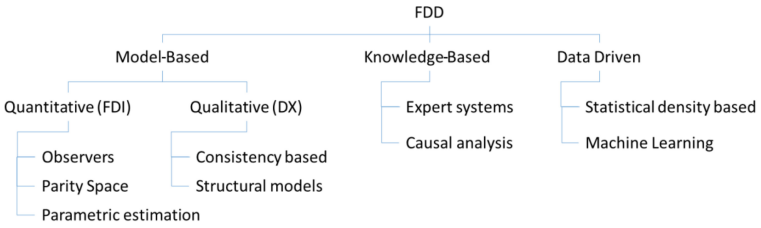

Figure 2.6: A taxonomy of Fault Detection and Diagnosis (FDD) methods for robotics. [127]

of sets called "constraint zonotopes" is introduced for set-based estimation and fault detection. An active, set-based FD method is developed in [131] based on convex polyhedrons to characterize the system's uncertainties with consistency checks of the outputs with unique fault models. Nevertheless, these approaches focus on linear or Linear Parameter Varying (LPV) systems.

Set Membership Estimation (SME) has been widely utilized for FD, offering a direct approach, by employing inverse tests for fault detection and concurrently estimating the feasible set from past input-output data. SME has the benefit that it can be extended to a larger class of systems, namely nonlinear systems that are linear to the parameters. Specific implementations of SME include zonotopic sets for fault detection, as demonstrated in [132, 133], and employing ellipsoids to delineate the parameter set, as seen in [134, 135]. Other studies, such as [136], apply state SME for FD. However, the works mentioned above, are limited to either linear and LPV systems or nonlinear systems but without considering both state and output uncertainties.

2.3.2 Fault Tolerance in Robotics

FT is the concept of containing the consequences of faults and failures to ensure a system maintains proper operation, even in the presence of errors. This is achieved primarily through redundancy in components and subsystems, which provides alternative paths for functionality when primary elements fail. FT algorithms typically work in tandem with Fault Diagnosis (FD) algorithms; the FD algorithms detect, isolate, and estimate the faults, providing detailed descriptions of any issues. The FT algorithms then use this information, combined with the system's redundancy, to devise and implement a recovery plan, thereby preserving the system's operational integrity.

Recently, there has been an increased interest in developing fault-tolerant systems to enhance safety and reliability across a wide range of applications. While fault tolerance is paramount in safety-critical tasks like navigation among human-operated vehicles, there are not many works that combine motion planning and fault tolerance since the first is usually seen as a higher-level task while the latter is usually combined with the model of the system at a lower level. In [137] a fault-tolerant steering control design for AVs is designed based on an adaptive state feedback controller. Actuator failures are also the focus of [138] where an adaptive fault-tolerant controller is proposed based on a proposed Lyapunov function to prove the stability of the adaptive control law. In [139] the authors focus on sensor faults where the latter are considered as additive signals estimated by a descriptor observer which considers the latter as a state variable of the vehicle model. An

adaptive sliding mode observer is designed in [140] to ensure the vehicle's safety when sensor faults in acceleration information exist. *This subsection is to be extended a bit after studying for Chapter 6.*

2.3.3 Fault Diagnosis and Fault Tolerance in Marine Systems

In the maritime domain, combinations of motion planning and fault-tolerant control are usually limited since FD is considered a lower-level component. Both in [141] and [142] the problem of path tracking is addressed with a fault tolerant control design to accommodate actuator faults. In [142] the authors consider time-varying multiplicative and additive actuator faults that are incorporated in the system's model and design a barrier Lyapunov function to prove that despite the presence of actuator faults and system uncertainties, the tracking errors converge to zero. FD in [143], an actuator fault-tolerant control scheme designed for an underwater Remotely Operated Vehicle (ROV) integrates detection, isolation, and accommodation modules. This work relies on residual generation modules for detection and exploits the specific actuator configuration for isolation through the sliding surface of a designed sliding mode controller [144]. The same ROV was studied in [145] where the authors focus on the problem of detection only, based on a nonlinear Thau observer for residual generation and on a sequential change detection algorithm for residual evaluation. A multiple sensor fault diagnosis scheme for ASVs is proposed in [146], utilizing various monitoring modules based on nonlinear observers to detect sensor faults. In addition, multiple fault isolation is achieved through a combinatorial decision logic approach, where the available sensors are grouped into multiple sensor sets. In [147] an active FD method is proposed for the same system so that actuator faults can be discerned from other disturbances by applying an auxiliary sinusoidal input system that is designed to propagate into the control system when a fault occurs while having minimal impact on the system dynamics. In [148] a bank of observers is used for FD in cascade with a nonlinear disturbance observer for fault estimation under the assumption that only a single fault may occur. Fault detection was studied for an underactuated surface vessel in [149] where a robust fault detection observer and a time-varying detection criterion are presented to detect the actuator faults distinguished from uncertainties and external disturbances. An Fault-Tolerant Control FTC strategy for linear systems is proposed in [120] with active FD that relies on the control redundancy of an overactuated ASV by constraining the inputs in prescribed configurations for Fault Detection, Isolation, and Reconfiguration (FDIR). However, this work relied on the linearization of vessel dynamics, assuming that the vessel's rotation is negligible with respect to translation motion, which might not hold in collision avoidance maneuvers.

2.4 Conclusions

The literature review highlights significant advancements and ongoing challenges in fault-tolerant motion planning within mixed-traffic environments, particularly focusing on AVs and ASVs. Motion planning encompasses route planning, behavioral decision-making, local motion planning, and feedback control, all of which are essential for enabling autonomous systems to navigate safely and efficiently. The review underscores the complexity of integrating these competencies due to uncertainties arising from traffic participants, sensor

noise, model inaccuracies, and external disturbances. A primary challenge in mixed-traffic environments is predicting the behaviors of other traffic participants, which remains inherently uncertain. Various probabilistic approaches have been proposed to address these uncertainties. Additionally, the review explores the importance of incorporating traffic rules into motion planning algorithms to enhance safety and predictability, highlighting both the successes and limitations of current methodologies. Considering traffic rules in motion planning is a promising way to simplify an inherently complex problem by introducing structure to the dynamic environment. While there is substantial work focusing on incorporating traffic rules into navigation algorithms for ASVs, there is a notable lack of research addressing this problem alongside ensuring the dynamic feasibility of the generated trajectories. Furthermore, existing studies often fall short of explicitly considering all relevant navigation rules and demonstrating scalability with multiple vessels (other traffic participants). Addressing these gaps is crucial for advancing the real-world deployment of autonomous navigation systems, ensuring that they can operate safely and effectively in diverse and dynamic environments.

Furthermore, fault-tolerant motion planning is identified as a crucial area of research to ensure the reliability and safety of autonomous systems. Faults, whether in sensors, actuators, or computational units, pose significant risks, necessitating robust mechanisms for FD and FT. The literature categorizes FD methods into model-based, signal-based, knowledge-based, hybrid, and process history-based approaches, each with unique advantages and limitations. However, the integration of FD and FT with motion planning remains limited, especially in marine environments where actuator faults and system uncertainties present additional challenges. Techniques such as adaptive state feedback controllers, sliding mode observers, and barrier Lyapunov functions have shown promise in accommodating faults and maintaining system stability. However, to the authors' knowledge, there are limited results on the integration of FD and FT modules that provide a reconfiguration strategy for the motion planner in case unexpected events such as faults occur.

Overall, while significant progress has been made in both rule-compliant and fault-tolerant motion planning, several critical gaps remain. Addressing the challenges of dynamic feasibility, comprehensive rule integration, and scalability in mixed-traffic environments, along with developing robust reconfiguration strategies for fault-tolerant motion planning, are essential for the advancement of autonomous navigation systems. This will be crucial for their successful real-world deployment and operation in increasingly complex and dynamic environments.

In the subsequent chapters, we focus on creating approaches that seamlessly integrate traffic rules, FD, and FT with motion planning to ensure the safety, reliability, and efficiency of autonomous systems in both urban and maritime contexts. Specifically, in Chapter 3, we develop an MPC-based trajectory optimization algorithm that considers marine traffic rules to generate feasible, rule-compliant trajectories for an ASV. Later, in Chapter 4, we develop an active FD algorithm based on residuals and adaptive thresholds to detect and isolate actuator (thruster) faults of the ASV. To identify the magnitude of these faults within certain margins, we then develop a more general FD method for nonlinear mechanical systems in Chapter 5, applying it to an ASV to robustly estimate thruster fault parameters along with a feasible parameter set. In Chapter 6, we combine the findings from Chapter 3 and Chapter 5 to derive a motion planning algorithm that can reconfigure itself according

to the system's health while handling traffic rule constraints, resulting in a rule-compliant and fault-tolerant motion planner. Finally, in Chapter 7, we summarize the findings of this thesis, highlight existing limitations, and propose promising research directions for future work.

3

Model Predictive Trajectory Optimization Considering Traffic Rules

*This chapter presents a rule-compliant trajectory optimization method for the guidance and control of ASVs as a response to Research Question **Q1**: "How can ASVs navigate safely and efficiently in dense traffic environments while ensuring compliance with maritime traffic rules?" The method builds on Model Predictive Contouring Control (MPCC) and incorporates the International Regulations for Preventing Collisions at Sea (COLREGs) relevant to motion planning. We use these rules for traffic situation assessment and to derive traffic-related constraints that are inserted in the optimization problem. Our optimization-based approach enables the formalization of abstract verbal expressions, such as traffic rules, and their incorporation in the trajectory optimization algorithm along with the dynamics and other constraints that dictate the system's evolution over a sufficiently long planning horizon. The ability to plan considering different types of constraints and the system's dynamics, over a long horizon in a unified manner, leads to a proactive motion planner that mimics rule-compliant maneuvering behavior, suitable for navigation in mixed-traffic environments. The efficacy and scalability of the derived algorithm are validated in different simulation scenarios, including complex traffic situations with multiple Obstacle Vessels (OVs). Section 3.2 describes the trajectory optimization problem. Section 3.1 presents a short introduction while Section 3.3 describes the vessel dynamics and Section 3.4 the path-following task. Decision-making based on the traffic rules is studied in Section 3.5 and the rule constraints are formulated in Section 3.6. Finally, Section 3.7 presents simulation results and Section 3.8 concludes the chapter.*

This chapter is based on **A. Tsolakis**, *RR Negenborn, V Reppa, and L Ferranti, "Model Predictive Trajectory Optimization and Control for Autonomous Surface Vessels Considering Traffic Rules", IEEE Transactions on Intelligent Transportation Systems, Feb 2024.*

3.1 Introduction

Over the past decade, we have witnessed the world of transportation rapidly advancing towards an increased level of automation. While the automotive industry has had the leading role in this trend, the maritime sector is also progressing towards developing and utilizing autonomous maritime systems in many applications including transportation [150], large-scale monitoring [151] or search and rescue missions [152]. Among the main societal benefits, the most interesting ones concern greater efficiency, reduced operational costs, and increased safety. According to [6], over the period 2014-2020, accidents of navigational nature (collisions, contacts, and groundings/strandings) represented almost 43% of all occurrences while human actions accounted for almost 61% of the contributing factors. Therefore, autonomous maritime navigation has the potential to significantly reduce the risk of collisions, which often lead to human casualties, damaged property, and devastating environmental disasters.

Despite the numerous benefits that autonomy has to offer in the maritime industry, the deployment of ASVs in real traffic environments is still limited. One of the main challenges to address relates to the transition period in which ASVs will be expected to co-exist with human-operated vessels in dense traffic environments, such as ports and inland waterways. This raises major societal concerns about the capabilities of the ASVs to interact safely with human-operated vessels in mixed-traffic conditions without causing disruptions or jeopardizing human safety. In this work, we propose a rule-compliant trajectory optimization and control method for ASVs that allows navigation in mixed-traffic environments.

In this chapter we extend the idea originally presented in [153] where we approached the problem of navigation in mixed-traffic environments by introducing a trajectory optimization algorithm for computing safe and rule-compliant trajectories for ASVs based on Model Predictive Contouring Control (MPCCs) since the latter has been proven to be especially suitable for autonomous vehicle applications [66, 154, 67, 155, 153, 112]. In contrast to other works that rely on heuristic hazard metrics and soft constraints for rule compliance, we rely on a purely geometric interpretation of the relevant rules and formulate hard constraints to enforce rule-compliant maneuvers while the vessel follows a time-invariant reference path. We formulate these constraints as affine expressions to keep the structure of the optimization problem simple and the algorithm scalable with respect to the number of Obstacle Vessels (OVs). While the collision-free space is generally nonconvex [156], the specific design of our constraints establishes a convex search space, encompassing homotopy-equivalent trajectories. Moreover, we leverage the predictive nature of the controller resulting in proactive, less conservative actions for the ASV while respecting the relevant traffic rules. Last but not least, we have also extended our work with respect to [153] by considering the dynamic model of the vessel including input and state constraints. The result is a trajectory optimization algorithm that achieves path following by generating dynamically feasible, rule-compliant, collision-avoiding trajectories within the prediction horizon while respecting actuator limitations as well. The contributions of this work are:

- A formal derivation of affine constraints that guarantees rule compliance in a convex search space.

- Simplified transition expressions in the traffic rule decision-making module that rely on the design of the affine constraints.
- An algorithm that scales to multiple obstacles and allows the vessels to safely navigate through dense traffic environments.

3.2 Problem Formulation

Consider that the ASV is moving in a planar workspace $\mathcal{W} = \mathbb{R}^2$. The motion is described by the discrete, nonlinear dynamical system:

$$\boldsymbol{x}(t+1) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \quad t = 0, 1, \dots, \tag{3.1}$$

with state $\boldsymbol{x} \in \mathcal{X}$ and control input $\boldsymbol{u} \in \mathcal{U}$ known by an appropriate set of sensors. We assume planar motion for n_o OVs as well, with their state defined as $\boldsymbol{x}^i \in \mathcal{X}^i$, $i = \{1, \dots, n_o\}$, known to sufficient precision within an area around the ASV along with an estimate of its length l^i and width w^i via a suitable perception framework [157, 158]. We take into account the subset of COLREGs rules 1-18 that describes navigation of vessels in *"sight of one another"*. The state of the ASV is constrained by these rules expressed mathematically as a set of state constraints denoted as $\mathcal{F}(\boldsymbol{x}_k, \boldsymbol{x}_k^i)$.

Given the current state $\boldsymbol{x}(t)$, a reference path parameterized by path parameter s initialized at $s(t)$, and a prediction of each OV's state $\boldsymbol{x}_{0:N|t}^i$, we formulate a discrete-time, constrained, receding horizon problem over a finite time horizon N with the set of states $\boldsymbol{x}_{0:N|t} \in \mathcal{X}$, set of inputs $\boldsymbol{u}_{0:N-1|t} \in \mathcal{U}$, and set of path parameters $s_{0:N}$ as decision variables:

$$\min_{\boldsymbol{x}_{\cdot|t}, \boldsymbol{u}_{\cdot|t}, s_{\cdot|t}} \sum_{k=0}^{N-1} J(\boldsymbol{x}_{k|t}, \boldsymbol{u}_{k|t}, s_{k|t}) + J_N(\boldsymbol{x}_{N|t}, s_{N|t}) \tag{3.2a}$$

$$\text{s.t.:} \quad \boldsymbol{x}_{k+1|t} = \boldsymbol{f}(\boldsymbol{x}_{k|t}, \boldsymbol{u}_{k|t}), \tag{3.2b}$$

$$s_{k+1|t} = g(\boldsymbol{x}_{k|t}, s_{k|t}), \tag{3.2c}$$

$$\boldsymbol{x}_{k|t} \in \mathcal{X} \cap \mathcal{F}(\boldsymbol{x}_{k|t}, \boldsymbol{x}_{k|t}^i), \tag{3.2d}$$

$$\boldsymbol{u}_{k|t} \in \mathcal{U}, \tag{3.2e}$$

$$\boldsymbol{x}_{0|t} = \boldsymbol{x}(t), \quad s_{0|t} = s(t), \tag{3.2f}$$

$$k = 0, \dots, N-1, \quad i = 0, \dots, n_o \tag{3.2g}$$

where we denote variables with subscript k as the predicted ones in the receding horizon problem. The solution to the receding horizon problem is the optimal input sequence $\boldsymbol{u}_{0:N-1}^*$ of the ASV that minimizes cost function (3.2a), under system dynamics (3.2b), path evolution (3.2c), state constraints (3.2d) and input constraints (3.2e). The cost function (3.2a) consists of the stage cost that is the sum of the following terms:

$$J(\boldsymbol{x}_{k|t}, \boldsymbol{u}_{k|t}, s_{k|t}) = \underbrace{J_v(\boldsymbol{x}_{k|t}) + J_{\boldsymbol{u}}(\boldsymbol{u}_{k|t})}_{\text{dynamic behavior}} + \underbrace{J_e(\boldsymbol{x}_{k|t}, s_{k|t}) + J_u(\boldsymbol{x}_{k|t})}_{\text{path following}} \tag{3.3}$$

and the terminal cost $J_N(\boldsymbol{x}_{N|t}, s_{N|t})$ that can be designed in order to ensure stability. The first two terms are designed to achieve a desirable dynamic behavior discussed in Section

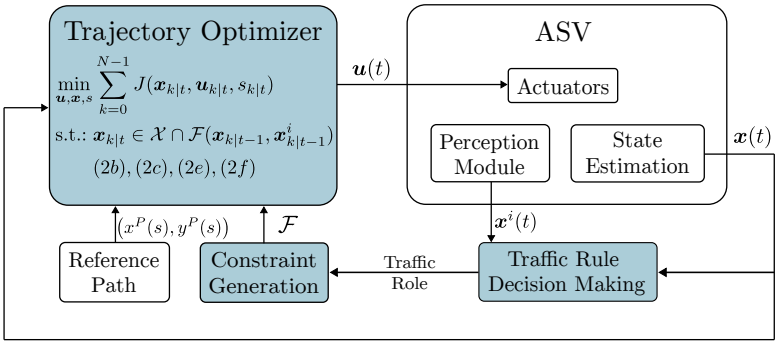

Figure 3.1: Schematic method overview (light blue blocks). Given the measured states $\boldsymbol{x}(t)$ and $\boldsymbol{x}^i(t)$, we first infer the traffic role of the vessels based on which a suitable set of constraints is generated. The latter is then inserted in the optimization problem and the first step of the optimal input sequence, $\boldsymbol{u}_1^*$, is applied to the ASV at each control cycle as $\boldsymbol{u}(t)$.

3.3 and the last two for navigation objectives discussed in Section 3.4. The dynamics (3.2b) and physical limitations of the state and inputs (3.2d), (3.2e) are detailed in Section 3.3 and the rule-compliance constraints (3.2d) that serve the task of rule-compliant collision avoidance are activated according to the decision-making scheme of Section 3.5 and are derived in Section 3.6. overview of our COLREGs-compliant navigation architecture is provided in Figure 3.1. We first encode the traffic rules in an algorithmic framework for situational awareness which is necessary for rule-compliant decision making. The module "Traffic Rule Decision Making" attributes a specific traffic role to the vessels based on which the "Constraint Generation" module generates a set of mathematical constraints that are suitable for a receding horizon problem and can guarantee a rule-compliant motion. The "Trajectory Optimizer" module then computes the trajectory for the vessel while considering the aforementioned constraints and outputs the corresponding control command to the ASV. Alternatively to previous works on MPCC [66, 154, 67, 155, 153, 112], we consider dynamic collision avoidance implicitly by enforcing compliance to the traffic rules.

We focus on the subset of the rules that are relevant to motion planning. They can be grouped into three categories: *Traffic Rule Decision Making* (7, 13-18) that analyze the situation and designate a traffic role to each vessel, *Situation Invariant Rules* (6, 8.a, 8.d) that apply irrespective of the traffic situation, and *Situation Dependent Rules* (8.b, 8.c, 8.e, 13-17) that vary according to the traffic role. The rest of the rules are either not implementable in motion planning (rules 1-5, 11, and 12) or can be better included in a higher-level motion planner that generates the reference path to be followed (rules 9 and 10).

3.3 Model Dynamics and Physical Limitations

For modeling vessel dynamics we rely on the maneuvering model described in [159]. The ASV's configuration is described by its position $\boldsymbol{p} = (x, y)^\top$, orientation ψ, longitudinal and lateral velocities u, v, and yaw rate r. Note that the velocities are expressed in the body reference frame of the vessel. We then denote as $\boldsymbol{x} = (x, y, \psi, u, v, r)^\top \in \mathcal{X} \subset \mathbb{R}^6$ the system's

state and as $\boldsymbol{u} = (\tau_l, \tau_r, \tau_b, \alpha_l, \alpha_r)^\top \in \mathcal{U} \subset \mathbb{R}^5$ the control input of an ASV with two azimuth thrusters at its beam and one bow thruster. Specifically, we denote as τ_l, τ_r, and α_l, α_r the thrusts and azimuths of the left and right azimuth thruster respectively, and as τ_b the thrust produced by a bow thruster of the ASV. Assuming that there are not any ocean currents, and wind or wave disturbances, the evolution of the system's state is expressed by the following continuous, nonlinear system:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \mathbf{0}_{3\times3} & \boldsymbol{R}(\boldsymbol{x}) \\ \mathbf{0}_{3\times3} & -\boldsymbol{M}^{-1}(\boldsymbol{C}(\boldsymbol{x}) + \boldsymbol{D}(\boldsymbol{x})) \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} \mathbf{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix} \boldsymbol{\tau}(\boldsymbol{u}), \tag{3.4a}$$

with:

$$\boldsymbol{M} = \boldsymbol{M}_{\boldsymbol{RB}} + \boldsymbol{M}_{\boldsymbol{A}}, \tag{3.4b}$$

$$\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{C}_{\boldsymbol{RB}}(\boldsymbol{x}) + \boldsymbol{C}_{\boldsymbol{A}}(\boldsymbol{x}), \tag{3.4c}$$

$$\boldsymbol{D}(\boldsymbol{x}) = \boldsymbol{D}_{\boldsymbol{L}} + \boldsymbol{D}_{\boldsymbol{NL}}(\boldsymbol{x}), \tag{3.4d}$$

$$\boldsymbol{\tau} = \begin{pmatrix} \tau_l \cos\alpha_l + \tau_r \cos\alpha_r \\ \tau_l \sin\alpha_l + \tau_r \sin\alpha_r + \tau_b \\ w_{lr}(\tau_r \cos\alpha_r - \tau_l \cos\alpha_l) - \\ \qquad l_{lr}(\tau_l \sin\alpha_l - \tau_r \cos\alpha_r) + l_b \tau_b \end{pmatrix} \tag{3.4e}$$

where $\boldsymbol{R}(\boldsymbol{x})$ is the rotation matrix, $\boldsymbol{M}_{\boldsymbol{RB}}$ the rigid-body mass matrix, $\boldsymbol{C}_{\boldsymbol{RB}}(\boldsymbol{x})$ the rigid-body Coriolis and centripetal matrix, $\boldsymbol{M}_{\boldsymbol{A}}$ the added-mass matrix, $\boldsymbol{C}_{\boldsymbol{A}}(\boldsymbol{x})$ the added Coriolis and centripetal matrix, $\boldsymbol{D}_{\boldsymbol{L}}$, $\boldsymbol{D}_{\boldsymbol{NL}}(\boldsymbol{x})$, the linear and nonlinear damping matrices, $\boldsymbol{\tau}$ the generalized force vector acting on the vessel, and w_{lr}, l_{lr}, l_b are length parameters that describe the configuration of the thrusters. The added-mass and Coriolis matrices are introduced due to hydrodynamic forces when we consider the additional forces resulting from the fluid acting on the vessel. The continuous system dynamics (3.4) are discretized with a Runge-Kutta method in the form (3.2b) to solve the receding horizon problem (3.2).

We also consider actuator limitations $\tau_l \in [\tau_{l_{\min}}, \tau_{l_{\max}}]$, $\tau_r \in [\tau_{r_{\min}}, \tau_{r_{\max}}]$, $\tau_b \in [\tau_{b_{\min}}, \tau_{b_{\max}}]$, $\alpha_l \in [\alpha_{l_{\min}}, \alpha_{l_{\max}}]$, $\alpha_r \in [\alpha_{r_{\min}}, \alpha_{r_{\max}}]$, where $\tau_{l_{\min}}$, $\tau_{l_{\max}}$, $\tau_{r_{\min}}$, $\tau_{r_{\max}}$, $\tau_{b_{\min}}$, $\tau_{b_{\max}}$, $\alpha_{l_{\min}}$, $\alpha_{l_{\max}}$, $\alpha_{r_{\min}}$, $\alpha_{r_{\max}}$ are the minimum and maximum control inputs respectively.

We can further include two terms in the objective function to tune the response of the dynamical system. First of all, to reduce undesirable drift of the vessel, we include the term:

$$J_v(\boldsymbol{x}_{k|t}) = q_v v_{k|t}^2, \quad k = 0, \dots, N-1, \tag{3.5}$$

to penalize lateral velocity v with tuning parameter q_v. Moreover, we penalize excessive control input by including the term:

$$J_{\boldsymbol{u}}(\boldsymbol{u}_{k|t}) = \boldsymbol{u}_{k|t}^\top \boldsymbol{Q}_{\boldsymbol{u}} \boldsymbol{u}_{k|t}, \quad k = 0, \dots, N-1, \tag{3.6}$$

where

$$\boldsymbol{Q}_{\boldsymbol{u}} = \begin{bmatrix} q_{\tau_l} & 0 & 0 & 0 & 0 \\ 0 & q_{\tau_r} & 0 & 0 & 0 \\ 0 & 0 & q_{\tau_b} & 0 & 0 \\ 0 & 0 & 0 & q_{\alpha_l} & 0 \\ 0 & 0 & 0 & 0 & q_{\alpha_r} \end{bmatrix}, \tag{3.7}$$

is a tuning parameter matrix. The rest of the states are subject to limitations imposed by the traffic rules as discussed in Section 3.6.

3.4 Path Following

The key idea in the MPCC problem formulation as expressed in (3.2), is that the vehicle does not need to track a reference trajectory but rather a time-invariant reference path via the objective function under certain input and state constraints. For the path following objective, we follow the approach in [66, 67] in which the vessel at time t is at position $\boldsymbol{p}(t) = (x(t), y(t))^\top$ and tracks a continuously differentiable two-dimensional reference path $(x^P(s), y^P(s))$ with path tangential angle $\psi^P(s) = \arctan(\partial y^P(s)/\partial x^P(s))$, parameterized by the arc length s. The arc length s of the closest point to the ASV can be approximated with an evolution of the path parameter (3.2c) described as:

$$s_{k+1|t} = s_{k|t} + u_{k|t}\Delta k, \tag{3.8}$$

with Δk denoting the prediction timestep, $u_{k|t}$ the discretized longitudinal velocity, and s_0 initialized at each planning cycle as the point of the path that is closest to the ASV's position. The path error vector $\boldsymbol{e}_{k|t}$ is then defined as:

$$\boldsymbol{e}_{k|t}(\boldsymbol{x}_{k|t}, s_{k|t}) = \begin{bmatrix} \tilde{e}^l(\boldsymbol{x}_{k|t}, s_{k|t}) \\ \tilde{e}^c(\boldsymbol{x}_{k|t}, s_{k|t}) \end{bmatrix}, \tag{3.9}$$

where the longitudinal error is defined as:

$$\tilde{e}^l(\boldsymbol{x}_{k|t}, s_{k|t}) = -\begin{pmatrix} \cos\psi^P(s_{k|t}) & \sin\psi^P(s_{k|t}) \end{pmatrix} \begin{pmatrix} x_{k|t} - x^P(s_{k|t}) \\ y_{k|t} - y^P(s_{k|t}) \end{pmatrix}, \tag{3.10}$$

and the contouring error as:

$$\tilde{e}^c(\boldsymbol{x}_{k|t}, s_{k|t}) = \begin{pmatrix} \sin\psi^P(s_{k|t}) & -\cos\psi^P(s_{k|t}) \end{pmatrix} \begin{pmatrix} x_{k|t} - x^P(s_{k|t}) \\ y_{k|t} - y^P(s_{k|t}) \end{pmatrix}, \tag{3.11}$$

To achieve path tracking using the definition of the error defined in (3.9), one of the cost terms in the objective function (3.2a) will take the form:

$$J_e(\boldsymbol{x}_{k|t}, s_{k|t}) = \boldsymbol{e}_{k|t}^\top \boldsymbol{Q}_{\boldsymbol{e}} \boldsymbol{e}_{k|t}, \quad k = 0, \dots, N-1, \tag{3.12}$$

where

$$\boldsymbol{Q}_{\boldsymbol{e}} = \begin{bmatrix} q_{e_l} & 0 \\ 0 & q_{e_c} \end{bmatrix}, \tag{3.13}$$

is a tuning parameter matrix that penalizes deviation from the reference path. A visual representation is illustrated in Figure 3.2.

To progress along the path, the ASV needs to have a non-zero longitudinal velocity $u_{k|t}$. This can be achieved by another term in the objective function:

$$J_u(\boldsymbol{x}_{k|t}) = q_u(u_{k|t} - u_{\text{ref}})^2, \quad k = 0, \dots, N-1, \tag{3.14}$$

where u_{ref} denotes a desired reference speed and q_u is a weighting factor to penalize deviation from the reference speed. Thus, the vessel can track a time-invariant path, the progress upon which is determined by the predicted longitudinal speed $u_{k|t}$. In this manner, the path-following task is quite flexible and allows the vessel to deviate from it if necessary (e.g., for collision avoidance) without creating conflicting objectives. The choice of these parameters q_u, u_{ref} is further discussed in Section 3.6 as it plays a role in rule compliance as well. For a more detailed description of the path following task the reader is referred to [66, 67].

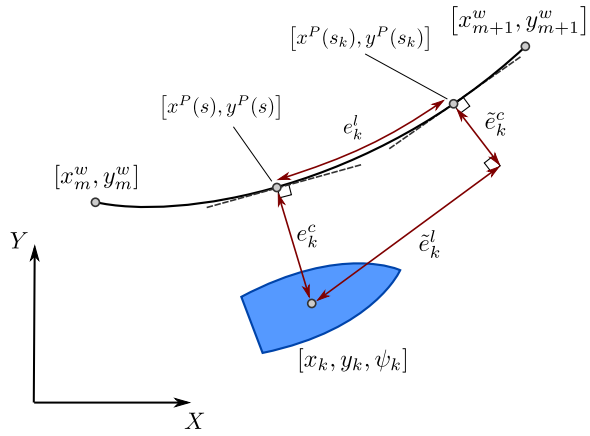

Figure 3.2: Path following with MPCC. A given reference path is considered and N reference points are chosen along its length depending on the longitudinal velocity of the ASV. The lag (longitudinal) and contouring (lateral) errors are minimized throughout the prediction horizon to achieve path following.

3.5 Traffic Rule Decision Making

Situation analysis and classification refers to a decision-making scheme that attributes a pairwise traffic role to the ASV and each OV, based on a subset of the traffic rules. This topic has been studied in great detail in [89, 110] among other works and is of great importance as it dictates the actions each vessel needs to follow in order to avoid collision in a safe manner. This section presents the simple Finite-State Machine (FSM) presented in Figure 3.3 that provides a pairwise role symmetry with transitions that consider properly defined entry and exit criteria for each state. The FSM has three states that represent the traffic role of the vessel - *Stand On* (SO), *Give Way* (GW), or *Emergency* (EM) as discussed in this Section. The corresponding transition expressions to enter or exit each state of the FSM namely, $T^{\text{ent}}_{\text{GW}}$, $T^{\text{ext}}_{\text{GW}}$, $T^{\text{ent}}_{\text{EM}}$, and $T^{\text{ext}}_{\text{EM}}$, depend on the current states $\boldsymbol{x}(t)$ and $\boldsymbol{x}^i(t)$ of the ASV and each OV and their derivation is presented step-by-step in this Section.

The first step is to identify if there exists risk of collision with an OV within the vicinity of the ASV. Rule 7 considers *"Risk of Collision"* with part 7.d.i describing that *"such risk shall be deemed to exist if the compass bearing of an approaching vessel does not appreciably change"* and part 7.d.ii *"such risk may sometimes exist even when an appreciable bearing change is evident, particularly when approaching a very large vessel or a tow or when approaching a vessel at close range"*. According to Rule 17.a.i, *"Where one of two vessels is to keep out of the way the other shall keep her course and speed"*. Thus, we can assume that any vessel encountered within an encounter radius denoted as ρ_{enc} around the ASV would keep a constant velocity if there is no risk of collision. We can then integrate the position vector equations from the current time t until some time in the future denoted as τ:

$$\boldsymbol{p}(\tau) = \boldsymbol{p}(t) + (\tau - t)\tilde{\boldsymbol{R}}(\boldsymbol{x}(t))\boldsymbol{v}(t), \tag{3.15a}$$

$$\boldsymbol{p}^i(\tau) = \boldsymbol{p}^i(t) + (\tau - t)\tilde{\boldsymbol{R}}(\boldsymbol{x}^i(t))\boldsymbol{v}^i(t), \tag{3.15b}$$

where we denote as $\boldsymbol{v} = (u, v)^\top$, $\boldsymbol{v}^i = (u^i, v^i)^\top$ the translational velocities of the two vessels

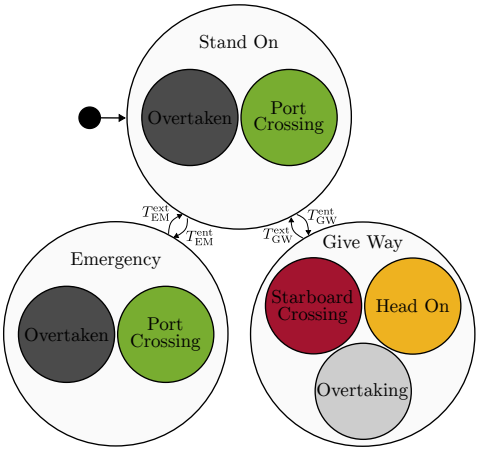

Figure 3.3: Schematic representation of the FSM for traffic role decision making. Traffic situations that lead to the same traffic role are grouped for simplicity.

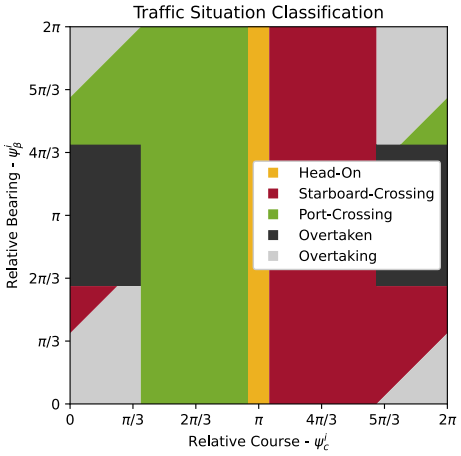

Figure 3.4: Traffic situation classification as a function of the relative course ψ_c^i, and the relative bearing ψ_β^i described in [82]. Note that the same figure from the perspective of the OV would be role-symmetric since pairwise-role symmetry is necessary.

and as $\tilde{\boldsymbol{R}}(\boldsymbol{x}(t))$ the 2×2 sub-matrix of $\boldsymbol{R}(\boldsymbol{x}(t))$, that maps the translational velocities from each body reference frame to the global reference frame. The current distance between the two vessels is:

$$d(t) = \left\| \boldsymbol{p}(t) - \boldsymbol{p}^i(t) \right\|_2 \tag{3.16}$$

The distance between two vessels at a future time τ computed at time t, can be expressed as:

$$d(\tau|t) = \left\| \boldsymbol{p}(t) - \boldsymbol{p}^i(t) + (\tau - t)(\tilde{\boldsymbol{R}}(\boldsymbol{x}(t))\boldsymbol{v}(t) - \tilde{\boldsymbol{R}}(\boldsymbol{x}^i(t))\boldsymbol{v}^i(t)) \right\|_2 \tag{3.17}$$

Both $d(t)$ and $d(\tau|t)$ are shown in Figure 3.5. Finding the minimum of $d(\tau|t)$ is equivalent to finding the minimum of its square, which is a quadratic function with respect to time τ. The minimum of this function is then the solution of $\partial d(\tau|t)^2/\partial\tau = 0$ which results to:

$$t_{\text{CPA}}(t) = -\frac{(\tilde{\boldsymbol{R}}(\boldsymbol{x}(t))\boldsymbol{v}(t) - \tilde{\boldsymbol{R}}(\boldsymbol{x}^i(t))\boldsymbol{v}^i(t))^\top(\boldsymbol{p}(t) - \boldsymbol{p}^i(t))}{\left\| \tilde{\boldsymbol{R}}(\boldsymbol{x}(t))\boldsymbol{v}(t) - \tilde{\boldsymbol{R}}(\boldsymbol{x}^i(t))\boldsymbol{v}^i(t) \right\|_2^2} \tag{3.18}$$

This future time is known as the time to the *"Closest Point of Approach"*. The corresponding distance is then:

$$d_{\text{CPA}}(t) = \begin{cases} \left\| \boldsymbol{p} - \boldsymbol{p}^i + (\tilde{\boldsymbol{R}}(\boldsymbol{x})\boldsymbol{v} - \tilde{\boldsymbol{R}}(\boldsymbol{x}^i)\boldsymbol{v}^i)t_{\text{CPA}} \right\|_2 & t_{\text{CPA}} \geq 0 \\ d & t_{\text{CPA}} < 0 \end{cases} \tag{3.19}$$

since $t_{\text{CPA}} < 0$ means that the two vessels are diverging and thus d_{CPA} is the current distance. Dependence on current time t is omitted for readability. We continue by assuming that a rough estimate of the length l^i and width w^i of the other vessel can be inferred by a visual perception or communication system (e.g., Automatic Identification System (AIS) and the

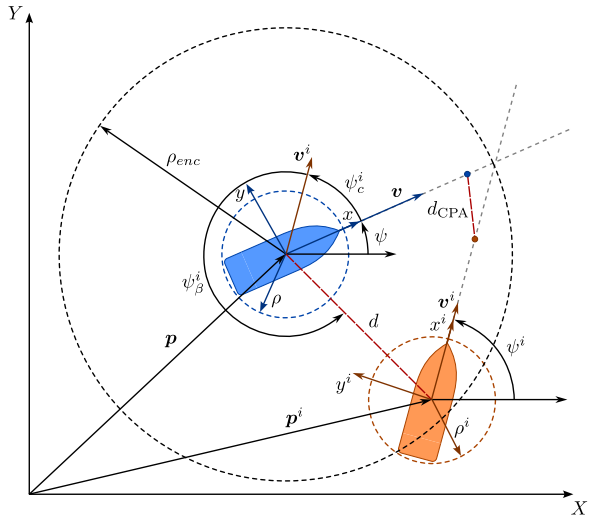

Figure 3.5: Encounter situation analysis between the ASV and OV i. Their current states $\boldsymbol{x}(t)$ and $\boldsymbol{x}^i(t)$ are used to determine the distance at the Closest Point of Approach (d_{CPA}) and thus if risk of collision exists assuming constant velocities.

footprint of vessels to be circles of radii $\rho = \sqrt{(l/2)^2 + (w/2)^2}$ and $\rho^i = \sqrt{(l^i/2)^2 + (w^i/2)^2}$, respectively, for the purposes of this module. Then, we can deem that risk of collision exists when $d_{\text{CPA}} < \rho + \rho^i + \rho_s$ which means that the two vessels will be closer than a safety margin ρ_s that for now is chosen arbitrarily. Note that using d_{CPA} as a metric for risk of collision is similar to the requirement of Rule 7 to monitor the relative bearing over time but easier to evaluate risk of collision considering the dimensions of the two vessels. For the purposes of traffic role decision making we use circular footprints as this serves only as a rough estimate of whether or not risk of collision exists. For collision avoidance we use a more accurate approximation of marine vessels' footprints discussed in Section 3.6.

If a risk of collision exists, the next step is to decide on the actions to be taken by the vessels involved. According to Rules 13-17, there can be three different, pair-wise traffic situations between the two vessels:

- Head-On / Head-On
- Starboard-Crossing / Port-Crossing
- Overtaking / Overtaken

These traffic situations depend on the relative position of the two vessels encoded in the *relative bearing*:

$$\psi_\beta^i(t) = \arctan\left(\frac{\hat{\boldsymbol{y}}^\top \tilde{\boldsymbol{R}}(\boldsymbol{x}(t))(\boldsymbol{p}^i(t) - \boldsymbol{p}(t))}{\hat{\boldsymbol{x}}^\top \tilde{\boldsymbol{R}}(\boldsymbol{x}(t))(\boldsymbol{p}^i(t) - \boldsymbol{p}(t))}\right), \tag{3.20}$$

and the *relative course*:

$$\psi_c^i(t) = \psi^i(t) - \psi(t), \tag{3.21}$$

with $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$ denoting the unit vectors of the ASV's body reference frame shown in Figure 3.5. The combination of $\psi^i_\beta(t)$ and $\psi^i_c(t)$ defines the role classification shown in Figure 3.4 similar to that found in [89]. To determine the head-on situation, we need to define one additional parameter ψ_h that defines a threshold for the relative course $\psi^i_c(t)$. Unfortunately, it is not clearly stated in the rules what the value should be but according to [110], court decisions indicate $\psi_h = \pm 6^\circ$. Note that for some combinations of $\psi^i_\beta(t)$ and $\psi^i_c(t)$ the traffic situations may not be considered if $d_{\text{CPA}} \geq \rho + \rho^i + \rho_s$ and risk of collision is not deemed to exist.

According to the rules, in each traffic situation, a vessel can be either a *Give-Way* (GW) vessel, which must take collision-avoiding action, or a *Stand-On* (SO) vessel, which is required to maintain its course and speed. According to this classification, each vessel has a GW or SO role as described in Rules 16 and 17 respectively. While Rule 16 is straightforward for the GW vessel, Rule 17.a.ii describes that *"The latter vessel"* (i.e., the SO) *"may however take action to avoid collision by her maneuver alone, as soon as it becomes apparent to her that the vessel required to keep out of the way is not taking appropriate action in compliance with these Rules"* and Rule 17.b states that *"When, from any cause, the vessel required to keep her course and speed finds herself so close that collision cannot be avoided by the action of the give-way vessel alone, she shall take such action as will best aid to avoid collision"*. Thus, another role emerges for the SO vessel which in some cases must take collision-avoiding action. We will denote this state here as *Emergency* (EM) state. This situation is studied in depth in [160] where they design a collision alert system for SO vessels. In summary, the following roles are expected from each vessel:

- GW: Head-On, Overtaking, and Starboard-Crossing
- SO: Port-Crossing and Overtaken with no needed action
- EM: Port-Crossing and Overtaken with emergency action

The last thing to consider for a complete encounter situation analysis is the entry and exit criteria. In [110] thresholds on d_{CPA} and t_{CPA} are defined in order to determine entry and exit criteria. However, these values may change rapidly especially in multi-vessel scenarios while the vessels are still in close proximity and likely to perform more complex maneuvers. Unfortunately, the rules do not describe explicitly for how long these pairwise roles should hold. Nevertheless, Rule 13.d clearly states that *"Any subsequent alteration of the bearing between the two vessels shall not make the overtaking vessel a crossing vessel within the meaning of these Rules or relieve her of the duty of keeping clear of the overtaken vessel until she is finally past and clear"*. Based on that we can infer that the pairwise roles, as long as they are attributed to the vessels, should remain consistent until the encounter situation is over. Thus, we keep the pairwise roles for as long as the other vessel remains within the encounter radius ρ_{enc} of the ASV for a normal traffic situation. An emergency situation is considered when $d < \rho_{\text{emg}}$ where ρ_{emg} defines the radius of a circular area around the ASV within which, if a GW vessel enters, it is inferred it does not comply with the rules. This is then deemed to be an emergency situation for which even as an SO vessel the ASV needs to take action to avoid collision according to Rule 17.

Lastly, in compliance with Rule 18.a, we assume that the perception system used by the ASV (e.g., similar to the one in [158]) can determine if the other vessel is *"(i) a vessel*

not under command; (ii) a vessel restricted in her ability to maneuver; (iii) a vessel engaged in fishing; (iv) a sailing vessel." which will set the role of the ASV to GW.

The aforementioned, lead to the design of the FSM illustrated in Figure 3.3 that is governed by the following Boolean expressions according to [82] that depend on the current states $\boldsymbol{x}(t)$ and $\boldsymbol{x}^i(t)$:

$$T_{\text{enc}} = d(t) < \rho_{\text{enc}} \tag{3.22a}$$
$$T_{\text{rsk}} = d_{\text{CPA}}(t) < \rho + \rho^i + \rho_s \tag{3.22b}$$
$$T_{\text{hdn}} = (\psi_c^i(t) \geq \pi - \psi_h) \wedge (\psi_c^i(t) < \pi + \psi_h) \tag{3.22c}$$
$$T_{\text{str}} = (\psi_c^i(t) \geq \pi + \psi_h) \wedge (\psi_c^i(t) < 13\pi/8) \tag{3.22d}$$
$$T_{\text{brn}} = (\psi_c^i(t) \geq 13\pi/8) \wedge (\psi_c^i(t) < 3\pi/8) \tag{3.22e}$$
$$T_{\text{ovr}} = (\pi + \psi_\beta^i(t) - \psi_c^i(t) \geq 5\pi/8) \wedge (\pi + \psi_\beta^i(t) - \psi_c^i(t) < 11\pi/8) \tag{3.22f}$$
$$T_{\text{stb}} = (\psi_\beta^i(t) \geq 0) \wedge (\psi_\beta^i(t) < 5\pi/8) \tag{3.22g}$$
$$T_{\text{emg}} = d(t) < \rho_{\text{emg}} \tag{3.22h}$$

which combined formulate the final transition expressions for the FSM of Figure 3.3:

$$T_{\text{GW}}^{\text{ent}} = T_{\text{enc}} \wedge \{T_{\text{rsk}} \wedge [T_{\text{hdn}} \vee T_{\text{str}} \vee (T_{\text{brn}} \wedge (T_{\text{ovr}} \vee T_{\text{stb}}))]\} \tag{3.23a}$$
$$T_{\text{GW}}^{\text{ext}} = \neg T_{\text{enc}} \tag{3.23b}$$
$$T_{\text{EM}}^{\text{ent}} = T_{\text{emg}} \tag{3.23c}$$
$$T_{\text{EM}}^{\text{ext}} = \neg T_{\text{emg}}, \tag{3.23d}$$

In the equations above, logic symbols $\wedge$, $\vee$, $\neg$, stand for "and", "or" and "not" respectively. Note that it is intentional that the EM state can only be reached from the SO state as we would like to allow vessels to come closer than ρ_{emg} if they adhere to the rules and they are assigned a pair of SO-GW roles. The FSM of Figure 3.3 can then assign the appropriate traffic role to each of the vessels. Note that for simplicity, the Overtaking, Head-On, and Starboard-Crossing situations have been grouped under the GW state and the Overtaken and Port-Crossing situations under the EM state, since the required actions are the same. Based on the traffic role assigned in this module, the corresponding collision avoidance constraints described in the next sections are generated and inserted in the optimization problem (3.2) before each planning cycle.

3.6 Constraint Generation

3.6.1 Situation Invariant Rules

The first rule that is implementable in a local motion planning algorithm is Rule 6, which describes that *"Every vessel shall at all times proceed at a safe speed so that she can take proper and effective action to avoid collision and be stopped within a distance appropriate to the prevailing circumstances and conditions"*. This rule is already implemented as a soft constraint in the cost function (3.2a) given in (3.14) as part of the path following task. The vessel's reference speed that needs to be followed can be set according to the local

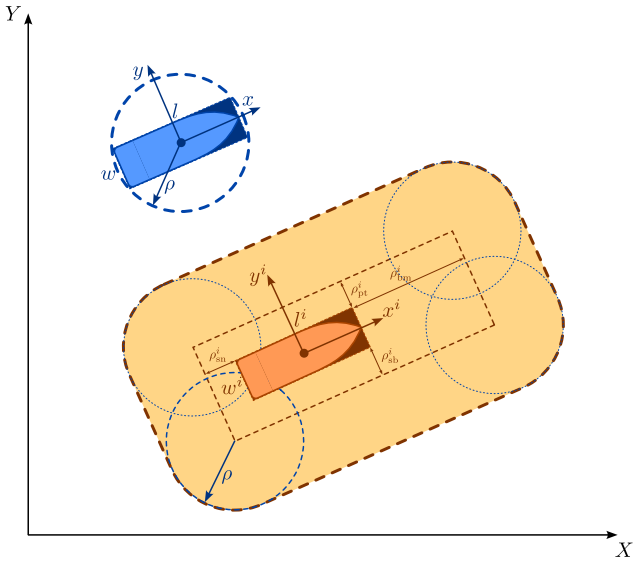

Figure 3.6: Footprints of the two vessels for collision avoidance. A rectangle shape is a simple representation of the real footprint of a vessel without being too conservative. The rectangle's sides can be augmented to allow for some safety margin as well. The rectangle is inflated by the radius of the circumscribed circle of the ASV leading to a rounded rectangle. Note that the ability to approximate the ASV with multiple circles of smaller radius can allow for less conservative approximations if needed (e.g. in inland waterways).

regulations that are applicable in its environment (e.g., open sea, canal, port, etc.) and the type of the vessel.

Rule 8 describes the proper action to avoid collision: Rule 8.a specifically describes that *"Any action to avoid collision [...], made in ample time [...]"*. This requirement is implemented with the already defined encountered distance ρ_{enc} between the two vessels which determines when the ASV has encountered another vessel and needs to assess the situation (see Figure 3.5).

Rule 8.d describes that action should be taken such that vessels are passing at a safe distance. While this is not explained adequately in the rules, we can think of what would be the best way to approximate the footprint of the OV. Because of the oblong shape that the vessels usually have, the circumscribed rectangle is a good approximation of the vessel's footprint since it is a simple shape but at the same time not very conservative (e.g., as the circumscribed circle would be). We can then implement this safety distance by enlarging the circumscribed rectangle by some margins $(\rho^i_{\text{bm}}, \rho^i_{\text{sn}}, \rho^i_{\text{pt}}, \rho^i_{\text{sb}})$ depending on the side of the vessel illustrated as the orange dashed rectangle in Figure 3.6. Since the decision variables include the center of the ASV where the body reference frame is attached, a common practice for the task of collision avoidance is to inflate the footprint of the obstacle by the dimensions of the ASV by using the Minkowski sum [161]. In general, the Minkowski sum depends on the relative orientation as well, which makes the computation of the inflated obstacle's footprint more involved, and the resulting shapes to vary. A simpler way is to approximate the footprint of the ASV with the circumscribed

circle which will make the Minkowski sum rotation-invariant. The Minkowski sum of the rectangular bound of vessel i and the circumscribed circle of the ASV with radius ρ is then the rounded orange rectangle illustrated in Figure 3.6, the most outer "boundary" around the OV. Notice that as done in previous works on MPCC [66, 154, 67], the footprint of the ego-vehicle (here the ASV) can be approximated with a multiplicity of offseted circles along the symmetry axis that will make the approximation much less conservative but still favorable in terms of computational complexity. This approximation of the OV's footprint with the safety margins is utilized in the following section where we generate the rule-compliant constraints.

3.6.2 Situation Dependent Rules

This section discusses rules that hold according to the encounter situation of the ASV. Rule 8.b states that *"Any alteration of course and/or speed to avoid collision shall, [...], be large enough to be readily apparent to another vessel [...]"*. This rule is often ignored leading to vessel maneuvers that are jittery and do not resemble rule-compliant maneuvers. One way to implement this rule is to impose constraints on the angular acceleration $\dot{r}$ and the longitudinal acceleration $\dot{u}$ to be larger than a certain value. However, as explained in [110], this can result in a highly non-convex (and even non-connected) search space and, consequently, in a hard-to-solve nonlinear optimization problem. Moreover, these variables are not included in (3.2). To circumvent these problems, we consider this rule in the design of constraints for Rules 13-17 later in this section. These constraints will cause the ASV to alter its course in a sufficient, rule compliant manner.

Rule 8.c states that *"If there is sufficient sea-room, alteration of course alone may be the most effective action to avoid a close-quarters situation [...]"*. This is already considered in (3.14) where we can tune weight q_u accordingly to track the reference speed. According to Rule 8.e, though, the vessel *"[...] shall slacken her speed or take all way off by stopping or reversing her means of propulsion"*. This means that the objective described in term (3.14) might interfere with collision avoidance as it then describes two conflicting goals for the trajectory optimizer. The problem can be overcome by switching the value of the tuning parameter q_u of cost term (3.14) according to the vessel role as $q_u \in \{q_{u_{\mathrm{SO}}}, q_{u_{\mathrm{GW}}}, q_{u_{\mathrm{EM}}}\}$ with $q_{u_{\mathrm{EM}}} \ll q_{u_{\mathrm{GW}}} = q_{u_{\mathrm{SO}}}$. Thus, in an emergency situation, the reference velocity following task is relaxed to allow the ASV to slow down or even reverse if necessary.

Next, we consider Rules 13-15, which describe the maneuver a GW vessel should follow in the *Overtaking*, *Head-On*, *Starboard Crossing* situations, respectively, as well as Rule 17 which describes emergency actions that arise in the *Overtaken* and *Port Crossing* situations for an EM vessel. Figure 3.7 presents examples of compliant (green) and non-compliant (red) maneuvers for each situation. In the following, we design suitable constraints to enforce compliant maneuvers while avoiding non-compliant ones.

In most MPC-based works these constraints are implemented as soft constraints via a heuristic cost function that relies on some hazard metric or aims at creating a repulsive field [103, 104, 105, 106, 107, 108, 109, 110, 111]. In this work, instead, the goal is to implement these rules as geometric, hard constraints to guarantee a rule-compliant behavior and decouple this task from the tasks of path-following and velocity-following described in the objective function. The design of these constraints should not cause problems with feasibility and allow the solution of (3.2) in real time. Thus, we design a set of affine

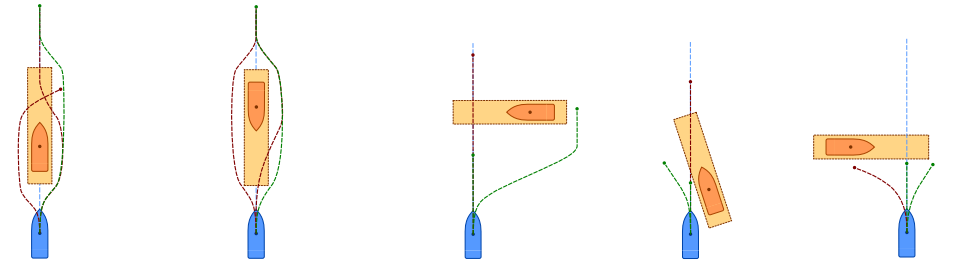

Figure 3.7: Rule-compliant (green) and prohibited (red) trajectories for the ASV (blue) in each traffic situation with an OV (orange) according to rules 13-17: The three situations on the left - a) Overtaking, b) Head On, c) Starboard Crossing - are situations where the ASV has a GW role while the two on the right - d) Overtaken, e) Port Crossing - describe suitable emergency maneuvers with the ASV in an EM role.

constraints for each pairwise situation. Then, in multi-vessel encounters, this will result to a convex polytope around the ASV, a rule-compliant search space in the receding horizon problem (3.2). We can then have strict rule-compliance guarantees in multi-vessel situations without complicating the solution of the optimization problem. These constraints might be more conservative than other types (e.g., quadratic constraints used in [66, 67]), but they are more suitable to represent the traffic rules as discussed in this Section. To design these constraints, we rely on the notion of the *separating* and *supporting* planes from convex optimization [162].

For each timestep $k \in 0,\dots,N-1$ along the prediction horizon, a supporting hyperplane of each circle $j \in [1,4]$ with radius ρ centered at the vertices of the inflated rectangle (see Figure 3.8) of OV $i \in 1,\dots,n$ can be defined as:

$$\mathcal{H}_d^{i,j} \; : \; \boldsymbol{d}_{k|t}^{i,j} \boldsymbol{p}_{k|t} \leq \boldsymbol{d}_{k|t}^{i,j\top} \left(\boldsymbol{p}_{k|t}^{i,j} + \boldsymbol{d}_{k|t}^{i,j} \rho \right), \tag{3.24}$$

where:

$$\boldsymbol{d}_{k|t}^{i,j} = \frac{\left(\hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j} \right)^\top}{\| \hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j} \|}, \tag{3.25}$$

is the normalized relative position vector defined with $\hat{\boldsymbol{p}}_{k|t}$ and $\boldsymbol{p}_{k|t}^{i,j}$ the predictions of the ASV and the OV's vertices respectively. For the OV we rely on a constant velocity assumption to derive the predicted positions as:

$$\boldsymbol{p}_{k|t}^{i} = \boldsymbol{p}^{i}(t) + k \cdot \Delta k \cdot \tilde{\boldsymbol{R}}(\boldsymbol{x}(t)) \boldsymbol{v}^{i}(t), \tag{3.26}$$

by inferring its current position $\boldsymbol{p}^{i}(t)$ and velocity $\boldsymbol{v}^{i}(t)$. The predicted positions of the

vertices are then:

$$\boldsymbol{p}_{k|t}^{i,1} = \boldsymbol{p}_{k|t}^{i} + \tilde{\boldsymbol{R}}(\boldsymbol{x}(t)) \begin{pmatrix} (l^i/2 + \rho_{\text{bm}}) \\ (w^i/2 + \rho_{\text{pt}}) \end{pmatrix} \tag{3.27a}$$

$$\boldsymbol{p}_{k|t}^{i,2} = \boldsymbol{p}_{k|t}^{i} + \tilde{\boldsymbol{R}}(\boldsymbol{x}(t)) \begin{pmatrix} (l^i/2 + \rho_{\text{bm}}) \\ -(w^i/2 + \rho_{\text{sb}}) \end{pmatrix} \tag{3.27b}$$

$$\boldsymbol{p}_{k|t}^{i,3} = \boldsymbol{p}_{k|t}^{i} + \tilde{\boldsymbol{R}}(\boldsymbol{x}(t)) \begin{pmatrix} -(l^i/2 + \rho_{\text{st}}) \\ -(w^i/2 + \rho_{\text{sb}}) \end{pmatrix} \tag{3.27c}$$

$$\boldsymbol{p}_{k|t}^{i,4} = \boldsymbol{p}_{k|t}^{i} + \tilde{\boldsymbol{R}}(\boldsymbol{x}(t)) \begin{pmatrix} -(l^i/2 + \rho_{\text{sn}}) \\ (w^i/2 + \rho_{\text{pt}}) \end{pmatrix} \tag{3.27d}$$

More general predictions from prediction modules can be accommodated as well. For the predictions of the ASV, we employ the trajectory of the previous planning cycle by shifting the previous plan one step forward: $\hat{\boldsymbol{p}}_k \triangleq \hat{\boldsymbol{p}}_{t|k} = \boldsymbol{p}_{t-1|k+1}$ for $k = 0, \dots, N-1$ while for the last step $k = N$ the predicted position is approximated as the linear extrapolation of the last two steps of the previous planning cycle: $\hat{\boldsymbol{p}}_N \triangleq \hat{\boldsymbol{p}}_{t|N} = 2\boldsymbol{p}_{t-1|N} - \boldsymbol{p}_{t-1|N-1}$. Note that the hyperplane of (3.24) can always be defined as long as $\hat{\boldsymbol{p}}_{k|t} \neq \boldsymbol{p}_{k|t}^{i,j}$ and is at the same time a separating hyperplane with respect to the ASV which is now reduced to a sequence of single points $\hat{\boldsymbol{p}}_{1:N}$ along the prediction horizon. Hyperplane $\mathcal{H}_d^{i,j}$, illustrated in Figure 3.8, can be used as a constraint to ensure that the footprints of the ASV and the OV will not overlap thus achieving collision avoidance. However, it cannot enforce rule-compliant trajectories similar to the green ones illustrated in Figure 3.7. For this reason, we want to rotate this hyperplane in a proper manner and force the generated trajectories as close to the desired ones as possible. That is, to the starboard side of the ASV and behind the OV as implicitly required by Rules 13-17. The range of rotation that keeps the supporting plane of each circle j to be a separating plane with respect to the ASV (each point $\hat{\boldsymbol{p}}_{k|t}$) is that between the two orange hyperplanes illustrated in Figure 3.8 denoted as $\mathcal{H}_d^{i,j}$ and $\mathcal{H}_{\text{max}}^{i,j}$. We are interested in the maximum counter-clockwise rotation of the orange hyperplane $\mathcal{H}_d^{i,j}$ with normal vector $\boldsymbol{d}_{k|t}^{i,j}$ that would lead to $\mathcal{H}_{\text{max}}^{i,j}$. The maximum angle of rotation $\theta_{k|t}^{i,j}$ is:

$$\theta_{k|t}^{i,j} = \begin{cases} \arccos\left(\frac{\rho}{\|\hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j}\|}\right) & \|\hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j}\| > \rho \\ 0 & \|\hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j}\| \leq \rho \end{cases} \tag{3.28}$$

Note that when $\theta_{k|t}^{i,j} = 0$ we have $\boldsymbol{r}_{k|t}^{i,j} = \boldsymbol{d}_{k|t}^{i,j}$ so that the rotated vector $\boldsymbol{r}_{k|t}^{i,j}$ can be defined even if $\|\hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j}\| \leq \rho$. Lastly, we introduce a rotation factor $\alpha \in [0, 1]$ as a tuning parameter with which we can tune the deflection of the predicted trajectory. The rotated vector is then:

$$\boldsymbol{r}_{k|t}^{i,j} = \tilde{\boldsymbol{R}}(\alpha\theta_{k|t}^{i,j})\boldsymbol{d}_{k|t}^{i,j} \tag{3.29}$$

The affine constraints will then take the form:

$$\mathcal{H}_r^{i,j} : \boldsymbol{r}_{k|t}^{i,j\,\top} \boldsymbol{p}_{k|t} \leq \boldsymbol{r}_{k|t}^{i,j\,\top} \left(\boldsymbol{p}_{k|t}^{i,j} + \boldsymbol{r}_{k|t}^{i,j}\rho\right) \tag{3.30}$$

Thus, the red hyperplane of Figure 3.8 denoted as $\mathcal{H}_r^{i,j}$ reduces smoothly to the orange hyperplane $\mathcal{H}_d^{i,j}$ as $\|\hat{\boldsymbol{p}}_{k|t} - \boldsymbol{p}_{k|t}^{i,j}\| \to \rho$ and the constraint can always be defined as long as

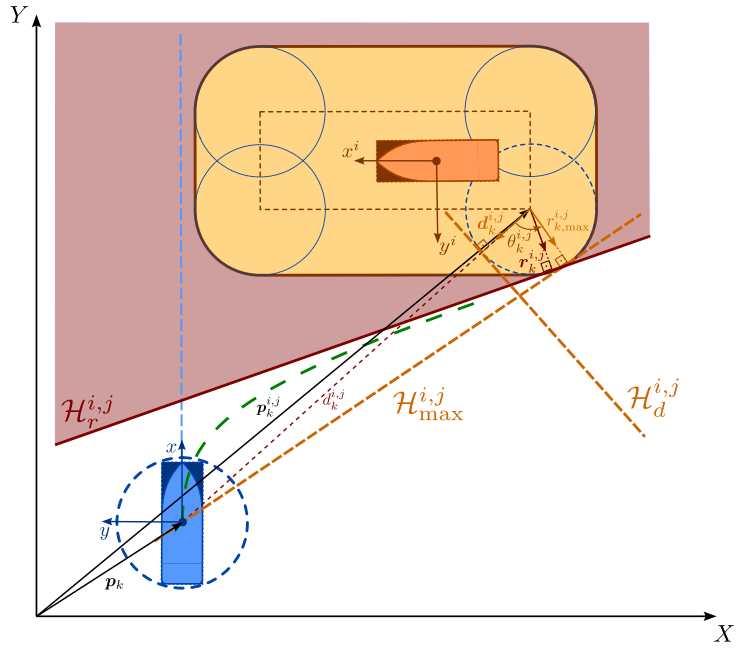

Figure 3.8: The affine constraint in the example of a starboard-crossing situation. The constraint restricts the allowable space for the ASV and forces the trajectory away from the reference path and behind the OV for the task of rule-compliant collision avoidance according to the rules.

$\hat{\boldsymbol{p}}_{k|t} \neq \boldsymbol{p}_{k|t}^{i,j}$. Depending on the traffic role of the ASV we can tune the rotation factor α differently to achieve a deflection of the trajectory as desired. In case the ASV has a GW role, in order to yield trajectories like the first three in Figure 3.7 this requires $\alpha \to 1$. On the other hand, if the ASV has an EM role, the last two trajectories of Figure 3.7 will be achieved for $\alpha \to 0$. If the ASV has an SO role, no constraints are imposed and the vessel is required to maintain its course and speed according to Rule 17. Lastly, in order to comply with Rule 8.b that requires readily apparent maneuvers, we can use the current states for the first few meters of the encounter that is, $\hat{\boldsymbol{p}}_{k|t} = \boldsymbol{p}_1$ and $\boldsymbol{p}_{k|t}^{i,j} = \boldsymbol{p}_1^{i,j}$. This will force a strong alteration of course or speed at the beginning of the encounter so that the actions of the ASV are readily apparent to the OVs.

Since these constraints are computed a priori based on the shifted plan, we can determine which one of them will be active and thus have only one constraint per obstacle to further simplify problem (3.2). Therefore, there will be a single constraint per OV that is "rolling" along the periphery of the rounded rectangle depending on the relative position and orientation of the ASV and the OV. In summary, each constraint i with $i = 1, \ldots, n$, splits the workspace of the vessels $\mathcal{W}$ in two half-spaces, one containing the i^{th} OV and its counterpart containing the ASV making sure that their footprints are always separated and thus collision avoidance is ensured. Moreover, the deflection tuning of this half-space is used to enforce rule-compliant trajectories. The affine constraints to be inserted in (3.2)

will then take the final form:

$$\mathcal{F} : \boldsymbol{L}_{k|t}^{o,i\top} \boldsymbol{p}_{k|t} \leq \boldsymbol{l}_{k|t}^{o,i}, \quad i = 1, \dots, n_o, \tag{3.31}$$

with:

$$\boldsymbol{L}_{k|t}^{o,i} = \boldsymbol{r}_{k|t}^{i,j}, \quad \boldsymbol{l}_{k|t}^{o,i} = \boldsymbol{L}_{k|t}^{o,i\top} \left(\boldsymbol{p}_{k|t}^{i,j} + \boldsymbol{L}_{k|t}^{o,i} \rho \right), \quad j \in [1,4] \tag{3.32}$$

where the index for $j \in [1,4]$ is chosen so that the corresponding affine constraint does not intersect the inflated rounded rectangle of Figure 3.8.

Note that the aforementioned considerations regarding i) a well-defined expression of constraints that ensures feasibility and ii) a pre-processing procedure to activate just one constraint per OV were not contemplated in [153]. In addition, in this work, a discussion on the effect of these constraints in traffic situations with multiple OVs follows. The constraint space $\mathcal{F}$ for the position $\boldsymbol{p}$ of the vessel is illustrated qualitatively in Figure 3.9. The ASV has either a GW or an EM role with respect to each OV and the corresponding affine constraint is generated. When these overlap, they lead to the convex polytope $\mathcal{F}$ which is the search space for the trajectory optimization problem (3.2). One of the benefits of such a design is that as the ASV and OV move with respect to each other to resolve the traffic situation, the constraints are "rolling" out of the way of the ASV thus not impeding its path anymore. Therefore, they can remain active for as long as the traffic role is active according to Section 3.5 without blocking the motion of the ASV. Thus, neither complicated exit criteria nor hysteresis in the decision-making module are needed for the FSM designed in Section 3.5 in contrast to other works (e.g., [87, 110]). Note that in Figure 3.9, the set $\mathcal{F}$ is presented only at the current time. In the optimization problem, there would be N polygons, one for each timestep k along the prediction horizon.

In the context of vessels navigating in dynamic environments with uncertain neighboring agents' intentions, ensuring formal closed-loop stability is challenging. One possible approach to address this issue is by modeling uncertainties in predicting neighboring vehicles' intentions within the motion planning problem and designing a suitable terminal cost in (3.2a) to ensure stability.

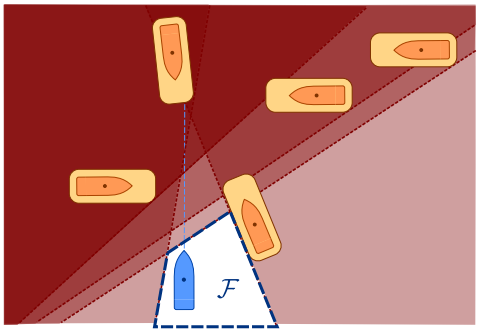

Figure 3.9: Multiple half-space constraints active simultaneously that result in a convex search space for the trajectory optimization problem.

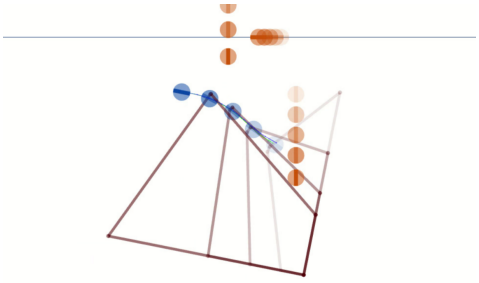

Figure 3.10: A screenshot from RVIZ to illustrate the convex polytope constraints generated along the prediction horizon for timestep $k = 10, 20, 30, 40$.

Algorithm 1 Traffic role decision making and constraint generation

Input: $\boldsymbol{x}(t), \boldsymbol{x}^i(t), l^i, w^i, \quad \forall i \in [1, \cdots, n]$
Output: $\boldsymbol{L}^o_{k|t}, \boldsymbol{l}^o_{k|t}, q_u$ see (3.31), (3.32)

for $t = 1, 2, \ldots$ **do**
  for $i \in [1, \cdots, n]$ **do** ▷ Traffic role decision making
    Compute $d(t)$ from Eq. (3.16)
    Compute $d_{\mathrm{CPA}}(t)$ from Eq. (3.19)
    Compute $\psi^i_b(t)$ from Eq. (3.20)
    Compute $\psi^i_c(t)$ from Eq. (3.21)
    Compute $role^i(t)$ from Eq. (3.23) and the FSM
  end for
  for $k \in [1, \cdots, N]$ **do** ▷ Constraint generation
    $\hat{\boldsymbol{p}}_k \leftarrow \boldsymbol{p}_{t-1|k+1}, \hat{\boldsymbol{p}}_N \leftarrow 2\boldsymbol{p}_{t-1|N} - \boldsymbol{p}_{t-1|N-1}$
    $\boldsymbol{p}^i_{k|t} \leftarrow \boldsymbol{p}^i(t) + k \cdot \Delta k \cdot \tilde{\boldsymbol{R}}(\boldsymbol{x}(t))\boldsymbol{v}^i(t)$
    for $i \in [1, \cdots, n]$ **do**
      if $role^i == SO$ **then**
        $q^i_u \leftarrow q_{u_{\mathrm{SO}}}$
      else
        if $role^i == GW$ **then**
          $q^i_u \leftarrow q_{u_{\mathrm{GW}}}$
          $\alpha \leftarrow [0, 1]$ ▷ Set α value close to 1
        else if $role^i == EM$ **then**
          $q^i_u \leftarrow q_{u_{\mathrm{EM}}}$
          $\alpha \leftarrow [0, 1]$ ▷ Set α value close to 0
        end if
        for $j \in [1, \cdots, 4]$ **do**
          Compute $\boldsymbol{p}^{i,j}_{k|t}$ from Eq. (3.27) given $\boldsymbol{p}^i_{k|t}$
          Compute $\boldsymbol{d}^{i,j}_{k|t}$ from Eq. (3.25) given $\hat{\boldsymbol{p}}_k$
          Compute $\theta^{i,j}_{k|t}$ from Eq. (3.28)
          Compute $\boldsymbol{r}^{i,j}_{k|t}$ from Eq. (3.29)
        end for
        Choose $\boldsymbol{r}^i_{k|t}$ as the active $\boldsymbol{r}^{i,j}_{k|t}$
        Compute $\boldsymbol{L}^{o,i}_{k|t}, \boldsymbol{l}^{o,i}_{k|t}$ from Eq. (3.31)
      end if
    end for
  end for
  $q_u \leftarrow min(q^i_u), \quad \forall i \in [1, \cdots, n]$
  Create $\boldsymbol{L}^o_{k|t}, \boldsymbol{l}^o_{k|t}$ by concatenating $\boldsymbol{L}^{o,i}_{k|t}, \boldsymbol{l}^{o,i}_{k|t}$
end for

3.7 Results

This section presents simulation results to validate the efficacy of our algorithm in different traffic scenarios. The first vessel-to-vessel scenarios are chosen to highlight the rule-compliant collision avoidance maneuvers in each possible traffic situation. We then test the algorithm in multi-vessel encounters to show that it does not lead to deadlocks in complex traffic situations and that it is scalable with respect to the number of OVs. Our framework is implemented in ROS: the controller in C++ and the simulator of the ASV and OVs in Python. The solver used relies on the Primal-Dual Interior-Point method and is generated with Forces Pro [163, 164]. The algorithm runs in an Ubuntu machine with an Intel i7 CPU@1.8GHz and 16GB of RAM.

In the following simulation scenarios, the ASV is expected to follow a horizontal reference path along the X-axis of the global reference frame at a reference surge velocity $u_{\text{ref}} = 1\text{m/s}$ while avoiding collisions according to the regulations. The values of the used parameters are summarized in Tables 3.1 and 3.2 while the numerical values of the ASV model described in (3.4a) can be found in [165]. For all the OVs the dimensions are the same as the ones used for the ASV: $l^i = l = 1.25\text{m}$ and $w^i = w = 0.29\text{m}$ while their longitudinal velocities vary in the range $0.9 - 1.2\text{m/s}$. The horizon length is set to $N = 41$ steps and the prediction timestep at $\Delta k = 0.25\text{s}$.

Figure 3.11 demonstrates the ASV's maneuver in an Overtaking situation where the ASV has a GW role. As described in Rule 13, the ASV turns to starboard while it keeps out of the way of the OV. Figure 3.12 shows the ASV in a Head-On situation and a GW role. In compliance with Rule 14, the ASV changes course to starboard so that each vessel passes on the port side of the other while it keeps out of the way of the OV.

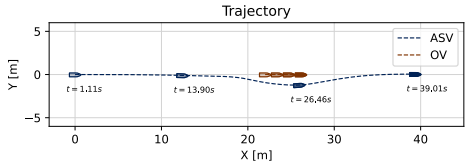

Figure 3.11: Overtaking situation with the ASV in GW role, turning to starboard while it keeps out of the way of the OV (Rule 13).

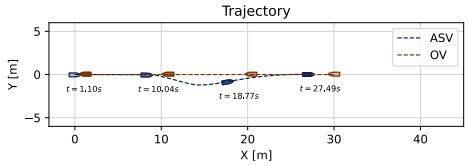

Figure 3.12: Head-On situation with the ASV in GW role, turning to starboard so that each vessel passes on the other's port (Rule 14).

Figure 3.13 illustrates another scenario in which the ASV has a GW role in a Starboard-Crossing situation. In this scenario, the ASV takes a collision avoidance maneuver to its starboard and avoids crossing ahead of the other vessel according to Rule 15. Lastly, Figure 3.14 presents a Port-Crossing situation where the ASV normally would have an SO role, but the OV does not comply with the rules and does not take action to avoid collision.

q_{e_l}	q_{e_c}	$q_{u_{\text{EM}}}$	$q_{u_{\text{GW}}}$	q_v	q_{τ_u}	q_{τ_r}
1	10	10	1000	250	0.1	3

Table 3.1: Objective function weight values

ρ_s	ρ_{enc}	ρ_{emg}	ψ_h	ρ^i_{bm}	ρ^i_{sn}	ρ^i_{pt}	ρ^i_{sb}	α
2	21	10	0.25	l^i	$l^i/2$	w^i	w^i	0.97

Table 3.2: Geometry parameter values

In this case, the ASV has an EM role and needs to take action to avoid collision while it does not alter its course to port for a vessel on its own port side. Notice that in every scenario, the ASV autonomously performs maneuvers that are clear and readily apparent thus complying with Rule 8.

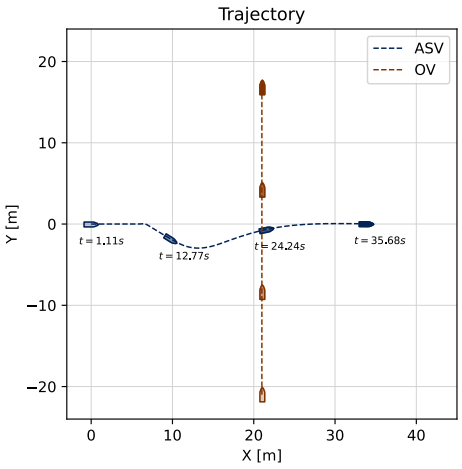

Figure 3.13: Starboard Crossing situation with the ASV in GW role, turning to starboard while avoiding crossing ahead of the other vessel (Rule 15).

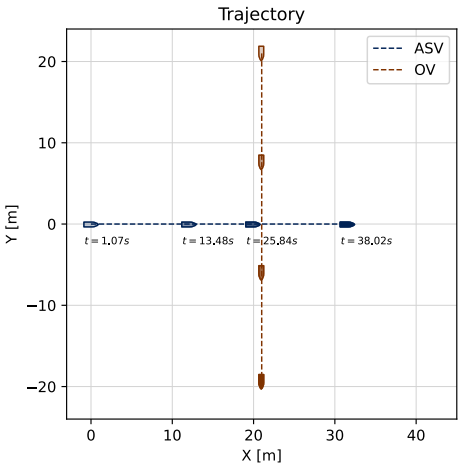

Figure 3.14: Port Crossing situation with the ASV in EM role, decelerating without turning to port passing behind the non-compliant vessel (Rule 17).

A multi-vessel encounter is illustrated in Figure 3.15 where the ASV is able to successfully avoid collision with each vessel obstacle in a rule-compliant manner. The ASV first encounters OV 1 and attempts to overtake it. A bit later it encounters two vessels (OV 2 and 3) crossing from its starboard side so it alters course to starboard to pass behind them. As soon as it returns to its reference path, OV 6 is coming from its port side not complying with the rules, and thus the ASV reduces speed to avoid collision. Right after, the ASV encounters OV 4 in a head-on situation and OV 5 in a port-crossing situation. At first, it changes course to starboard and later slows down to successfully avoid collision with both, according to the rules.

The simulation environment in which we run our experiments (RVIZ) is illustrated in Figure 3.10 where the constraint polytopes can be seen along the prediction horizon. The corresponding state and input of the system for the multi-vessel scenario is provided in Figures 3.18, 3.19, and 3.20. In Figure 3.16 the successive traffic roles are shown as the ASV navigates through traffic. The ASV has a GW role with respect to OV 1, 2, 3 and 4 and an SO role with respect to OV 5 and 6. The latter do not comply with the rules and thus an

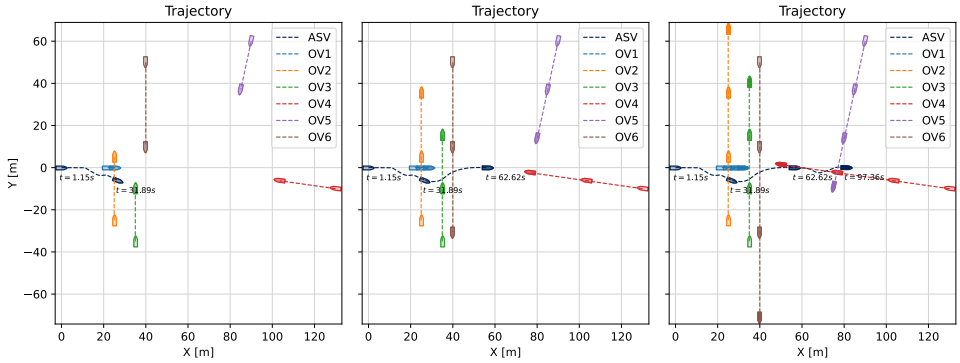

Figure 3.15: Trajectories in a multi-vessel encounter situation with the ASV passing through multiple OVs while following a horizontal path.

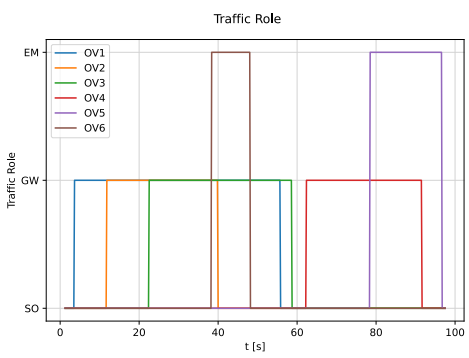

Figure 3.16: Traffic role for the ASV with respect to each OV corresponding to the traffic situations that emerge in Figure 3.15.

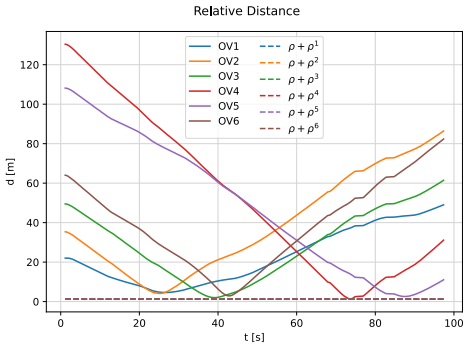

Figure 3.17: The distance between the ASV and each OV in the scenario presented in Figure 3.15 as a function of time. The lower dashed line represents the sum of $\rho+\rho^i$ for each OV showing that there is no collision.

EM role emerges for the ASV as they approach in dangerous proximity. In Figure 3.17 we compare the relative distance between the ASV and each OV i to the minimum accepted distance for collision avoidance ($\rho+\rho^i$). Note that this is more conservative than what we enforce with the collision avoidance constraints, but it is used just as an indication that collision avoidance is achieved. Lastly, we show in Figure 3.21 the computation time with respect to the increasing number of obstacles to illustrate the scalability of the algorithm. The average time for the control loop is on average about 33 ms for every run showing that the number of obstacles does not complicate the solution of the optimization problem.

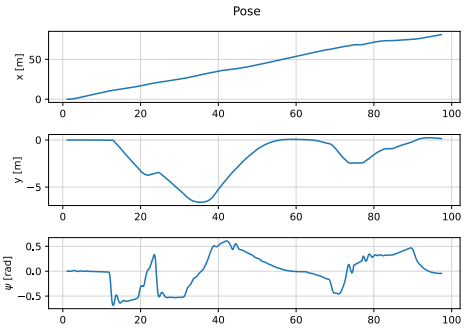

Figure 3.18: Pose (position and orientation) of the ASV for the multi-vessel encounter presented in Figure 3.15.

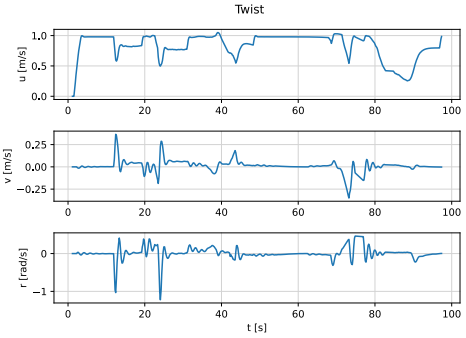

Figure 3.19: Twist of the ASV for the multi-vessel encounter presented in Figure3.15.

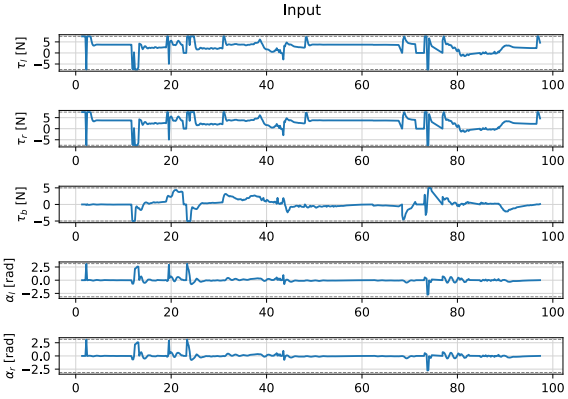

Figure 3.20: Control input of the ASV for the multi-vessel encounter presented in Figure 3.15.

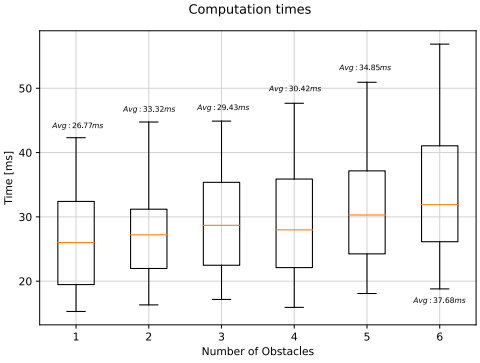

Figure 3.21: Computation times for an increasing number of obstacles: The average computation time remains similar meaning that the additional constraints do not complicate the solution of the optimization problem showcasing the scalability of the method.

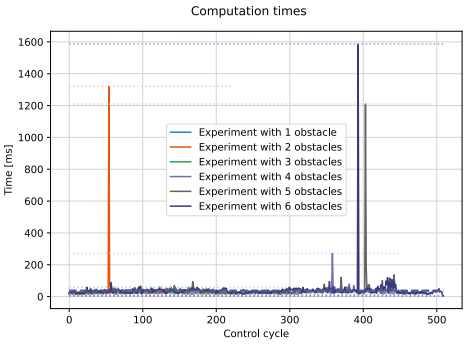

Figure 3.22: The computation times with respect to the control cycle for an increasing number of obstacles. While the average time stays within acceptable limits, there are edge cases that complicate the application in real-time.

3.8 Conclusions

In this chapter, we proposed a trajectory optimization and control algorithm for safe navigation of ASVs in mixed-traffic environments, that is, environments with human-operated vessels by incorporating COLREGs as constraints as a response to Research Question **Q1**: "How can ASVs navigate safely and efficiently in dense traffic environments while ensuring compliance with maritime traffic rules?" The efficacy of the proposed algorithm was validated via different simulation scenarios involving relevant rule-compliant collision avoidance maneuvers that comply with COLREGs. Scenarios with multiple vessels were also tested to show the algorithm's ability to handle complex traffic situations without deadlocks and its scalability with respect to the number of obstacles. In the following chapters, we shift our focus to addressing uncertainties and faults by designing suitable FD and FTC modules to complement the motion planner, ensuring reliable operation even in the presence of disturbances and system anomalies.

4

Active Thruster Fault Diagnosis

As ASVs become increasingly prevalent in marine applications, ensuring their safe operation, in the presence of faults, is crucial to human safety. This chapter addresses Research Question ***Q2****: "How to detect and isolate actuator faults in ASVs to enhance overall operational safety and reliability?" by presenting a scheme that encompasses the detection and isolation of actuator faults for ASVs to ensure uninterrupted and safe operation. The method primarily addresses the loss of thruster effectiveness as a specific actuator fault. For fault detection, the proposed method leverages residuals generated by nonlinear observers, coupled with adaptive thresholds, enhancing fault detection accuracy. The active fault isolation strategy employs actuator redundancy to insulate specific system states from faults by dynamically reconfiguring the actuation configuration in response to detected faults. Comprehensive simulation results demonstrate the effectiveness of this methodology across diverse marine traffic scenarios where the ASV needs to perform a collision avoidance maneuver. This chapter is organized as follows: Section 4.1 briefly introduces the overall idea. Section 4.2 describes the formulated FDI problem for a 3-DoF ASV under environmental disturbances, measurement noise, and specific actuator faults. Section 4.3 details the fault diagnosis method, which involves a cascaded detection and isolation procedure. Finally, Section 4.4 presents simulation results and Section 4.5 concludes this chapter with some additional remarks.*

This chapter is based on 🖹 ***A. Tsolakis****, L. Ferranti, and V. Reppa, "Active Thruster Fault Diagnosis for an Overactuated Autonomous Surface Vessel", Proc. of the 12th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS 2024), June 2024.*

4.1 Introduction

In recent years, there has been a strong interest in developing autonomous solutions for marine systems, spanning various applications. These include for example autonomous surface vessels in the transportation of passengers and goods, unmanned surface vessels for environmental monitoring and bathymetric mapping, and autonomous underwater vehicles employed in tasks such as exploration and inspection of underwater structures. While these constitute promising, cost-effective solutions that could enhance efficiency, there are still concerns regarding the safe operation and reliability of these systems especially in environments shared with other human-operated vehicles.

In the previous chapter, we introduced a trajectory optimization method that allows ASVs to navigate among other human-operated vessels by complying with the traffic rules. While the method proved to be effective in nominal, *healthy* conditions, the critical question remains on how safety can be guaranteed even in the presence of *faults*.

As autonomous vehicles heavily rely on components such as sensors, actuators, computation units, and various other systems, a major concern revolves around the potential consequences of component faults or complete failures during operation. Recent research has predominantly focused on FTC for these systems, aiming to maintain system functionality or ensure safety despite the occurrence of faults or failures. This chapter focuses on FD, a critical component of FTC that aims to enhance the system's health understanding. Our work highlights the importance of accurately identifying and localizing faults, thereby improving the safety and reliability of autonomous marine vessels.

This chapter presents an active FDI scheme designed to complement the rule-compliant trajectory optimization algorithm for ASVs proposed in Chapter 3 and [166]. While previous works primarily focus on passive fault diagnosis or rely on fixed detection thresholds, we introduce a planning-integrated, adaptive FD method capable of detecting and isolating actuator faults in real time. This enhances overall system safety by proactively addressing actuator failures within the motion planning framework. In contrast to conventional methods that use static or heuristic-based thresholds for fault detection, we derive adaptive thresholds that dynamically adjust based on system nonlinearities. This improves detection accuracy while simultaneously accounting for bounded noise and disturbances. Additionally, fault isolation is achieved through the inherent control redundancy of the vessel and the explicit representation of model dynamics and input constraints within the MPC formulation of [166]. This eliminates the need for auxiliary control allocation modules and avoids unnecessary input saturation handling, simplifying the overall architecture. The contributions of this work are:

- A novel adaptive fault detection scheme that dynamically adjusts detection thresholds based on system nonlinearities, improving accuracy while accounting for bounded noise and disturbances.
- A fault isolation approach that leverages control redundancy and integrates seamlessly with the MPC framework, removing the need for auxiliary control allocation and saturation handling.
- A planning-integrated fault diagnosis strategy that proactively accounts for actuator faults, ensuring safer and more reliable motion planning in mixed-traffic maritime environments.

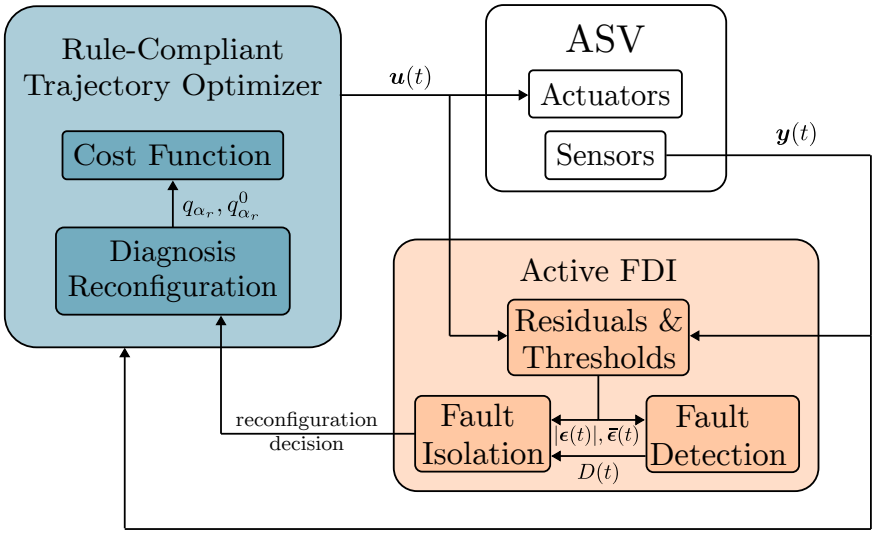

Figure 4.1: Schematic method overview (light orange block). Fault detection and isolation are realized given the input $\boldsymbol{u}(t)$ and measurement $\boldsymbol{y}(t)$.

4.2 Problem Formulation

We consider the ASV dynamics as a 3-DoF system in planar motion. We assume that the ASV is already equipped with a set of sensors and actuators as well as the trajectory optimizer developed in [166] for rule-compliant collision avoidance (highlighted in blue in Figure 4.1). In this work, we focus on developing the "FDI" block, highlighted in orange in Figure 4.1, that utilizes the system's input and output data.

The vessel dynamics are described by the maneuvering model in [159]. The ASV's configuration is described by its position $\boldsymbol{p} = (x, y)^\top$, orientation ψ, longitudinal and lateral velocities u, v, and yaw rate r. Note that the velocities are expressed in the body reference frame of the vessel. We then denote as $\boldsymbol{x} = (x, y, \psi, u, v, r)^\top \in \mathcal{Z} \subset \mathbb{R}^6$ the system's state and as $\boldsymbol{u} = (\tau_l, \tau_r, \tau_b, \alpha_l, \alpha_r)^\top \in \mathcal{U} \subset \mathbb{R}^5$ the control input of an overactuated ASV with two azimuth thrusters at its beam and one bow thruster. Specifically, we denote as τ_l, τ_r, and α_l, α_r the thrusts and azimuths of the left and right azimuth thruster respectively, and as τ_b the thrust produced by the bow thruster of the ASV. Environmental disturbance forces from the wind and waves are denoted as $\boldsymbol{\tau_d} \in \mathcal{D} \subset \mathbb{R}^3$. The evolution of the system's state

is expressed by the following continuous, nonlinear system:

$$\dot{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \mathbf{0}_{3\times3} & \boldsymbol{R}(\boldsymbol{x}) \\ \mathbf{0}_{3\times3} & -\boldsymbol{M}^{-1}(\boldsymbol{C}(\boldsymbol{x})+\boldsymbol{D}(\boldsymbol{x})) \end{bmatrix} \boldsymbol{x}}_{f(\boldsymbol{x})} + \underbrace{\overbrace{\begin{bmatrix} \mathbf{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix}}^{\tilde{\boldsymbol{M}}} \boldsymbol{\tau}(\boldsymbol{u})}_{g(\boldsymbol{u})} + \underbrace{\overbrace{\begin{bmatrix} \mathbf{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix}}^{\tilde{\boldsymbol{M}}} \boldsymbol{\tau}_{\boldsymbol{d}}}_{\boldsymbol{d}} \tag{4.1a}$$

with:

$$\boldsymbol{M} = \boldsymbol{M}_{RB} + \boldsymbol{M}_{A}, \tag{4.1b}$$

$$\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{C}_{RB}(\boldsymbol{x}) + \boldsymbol{C}_{A}(\boldsymbol{x}), \tag{4.1c}$$

$$\boldsymbol{D}(\boldsymbol{x}) = \boldsymbol{D}_{L} + \boldsymbol{D}_{NL}(\boldsymbol{x}), \tag{4.1d}$$

$$\boldsymbol{\tau}(\boldsymbol{u}) = \begin{pmatrix} \theta_l \tau_l \cos\alpha_l + \theta_r \tau_r \cos\alpha_r \\ \theta_l \tau_l \sin\alpha_l + \theta_r \tau_r \sin\alpha_r + \theta_b \tau_b \\ w_{lr}(\theta_r \tau_r \cos\alpha_r - \theta_l \tau_l \cos\alpha_l) - \\ \quad l_{lr}(\theta_l \tau_l \sin\alpha_l - \theta_r \tau_r \cos\alpha_r) + l_b \theta_b \tau_b \end{pmatrix} \tag{4.1e}$$

where $\boldsymbol{R}(\boldsymbol{x})$ is the rotation matrix, $\boldsymbol{M}_{RB}$ the rigid-body mass matrix, $\boldsymbol{C}_{RB}(\boldsymbol{x})$ the rigid-body Coriolis and centripetal matrix, $\boldsymbol{M}_{A}$ the added-mass matrix, $\boldsymbol{C}_{A}(\boldsymbol{x})$ the added Coriolis and centripetal matrix, $\boldsymbol{D}_{L}$, $\boldsymbol{D}_{NL}(\boldsymbol{x})$, the linear and nonlinear damping matrices, $\boldsymbol{\tau}$ the generalized force vector acting on the vessel, and w_{lr}, l_{lr}, l_b are length parameters that describe the configuration of the thrusters. The added-mass and Coriolis matrices are introduced due to hydrodynamic forces when we consider the additional forces resulting from the fluid acting on the vessel. The thrust force from the actuators in healthy conditions is denoted as $\boldsymbol{\tau}(\boldsymbol{u})$ with $\{\theta_l, \theta_r, \theta_b\}$ fault parameters described at the end of this section. Actuator limitations are considered as well. The actuators' configuration is illustrated in Figure 4.2.

We model disturbances based on [167] where the prevailing disturbance force is due to the wind, and then wave and current disturbance forces are due to the wind forces. We assume that this disturbance is unknown but with a known upper bound denoted as $\bar{\boldsymbol{\tau}}_{\boldsymbol{d}}$. We, therefore, model disturbance as a truncated Gaussian random variable with mean $\boldsymbol{\mu}_{\boldsymbol{d}}$ and variance $\Sigma_{\boldsymbol{d}}$:

$$\boldsymbol{d} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{d}}, \Sigma_{\boldsymbol{d}}) \quad \text{for} \quad \boldsymbol{\mu}_{\boldsymbol{d}} - 2\Sigma_{\boldsymbol{d}} \leq \boldsymbol{\tau}_{\boldsymbol{d}} \leq \boldsymbol{\mu}_{\boldsymbol{d}} + 2\Sigma_{\boldsymbol{d}} = \bar{\boldsymbol{d}} \tag{4.2}$$

We assume that we have access to a full-state measurement that is corrupted by a noise signal $\boldsymbol{n}$ that is unknown but bounded with the bound denoted as $\bar{\boldsymbol{n}}$:

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n} \tag{4.3}$$

We model noise measurement as a zero-mean truncated Gaussian random variable with variance $\Sigma_{\boldsymbol{n}}$:

$$\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \Sigma_{\boldsymbol{n}}) \quad \text{for} \quad -2\Sigma_{\boldsymbol{n}} \leq \boldsymbol{n} \leq 2\Sigma_{\boldsymbol{n}} = \bar{\boldsymbol{n}} \tag{4.4}$$

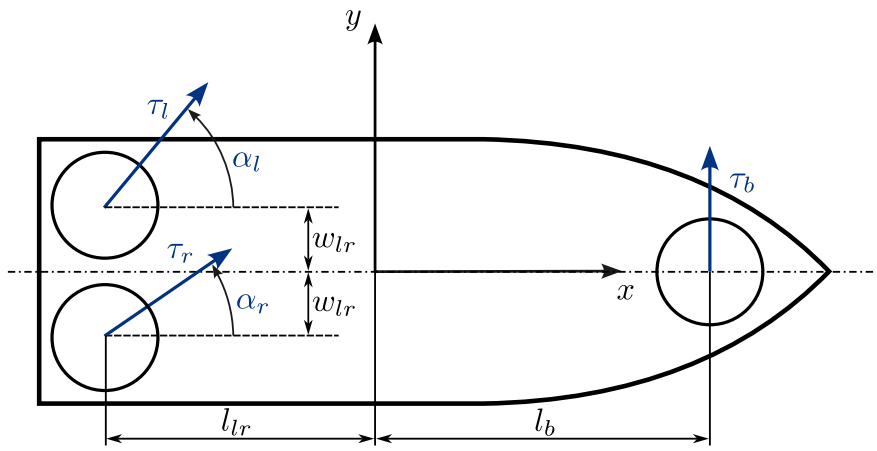

Figure 4.2: Schematic representation of the actuators' configuration with two azimuth thrusters at the stern and one bow thruster.

Lastly, for fault modeling, we consider actuator faults and more specifically thruster loss of effectiveness (LoE), which is widely considered as a relevant actuator fault [120, 168, 149, 148]. The fault parameters in (4.1e) are then given as:

$$\theta_j = \begin{cases} 1, & t < t_{f_i} \\ 0 < \theta_j < 1, & t \geq t_{f_i} \end{cases} \tag{4.5}$$

For healthy conditions, we have $\theta_j = 1$ while $\theta_j = 0$ means complete failure. Time instant t_{f_i} denotes the time a fault occurs. We further assume that the fault happens abruptly after fault time t_{f_i} and that only single faults occur since in practice it is infrequent that two or more actuator faults occur simultaneously ([148, 169]). We also assume that there are no sensor faults affecting the system.

The goal of this work is to develop a scheme that can detect and isolate parameters $\theta_j, i \in \{l, r, b\}$ when these deviate from healthy conditions (i.e., when $\theta_j \neq 1$) under disturbances (4.2) and measurement noise (4.4).

4.3 Active Fault Diagnosis based on Residuals and Adaptive Thresholds

4.3.1 Residuals & Thresholds

The system equations (4.1) and (4.3) can be re-written in compact form as:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{u}) + \boldsymbol{d} \tag{4.6a}$$

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n} \tag{4.6b}$$

We design a nonlinear observer to generate residuals and the respective adaptive thresholds. The nonlinear observer can be expressed as:

$$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{f}(\hat{\boldsymbol{x}}) + \boldsymbol{g}^H(\boldsymbol{u}) + \boldsymbol{\Lambda}(\boldsymbol{y} - \hat{\boldsymbol{y}}) \tag{4.7a}$$

$$\hat{\boldsymbol{y}} = \hat{\boldsymbol{x}} \tag{4.7b}$$

where $\hat{\boldsymbol{x}}$ is the state estimate vector and $\boldsymbol{\Lambda}$ is the observer gain which is a positive definite diagonal matrix and $\boldsymbol{g}^H(\boldsymbol{u})$ denotes the input map in healthy conditions i.e., when there are no actuator faults and $\eta_j = 1,\ \forall j \in \{l, r, b\}$. The residual is expressed as:

$$\boldsymbol{\epsilon} = \boldsymbol{y} - \hat{\boldsymbol{y}} \tag{4.8}$$

Substituting (4.6b) to (4.8) and using the triangle inequality we get:

$$|\underbrace{\boldsymbol{y} - \hat{\boldsymbol{y}}}_{\text{residual}}| \leq |\tilde{\boldsymbol{x}}| + \bar{\boldsymbol{n}} \tag{4.9}$$

with $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \hat{\boldsymbol{x}}$ the state error. Note that inequalities between matrices are to be interpreted element-wise where $|\cdot|$ denotes the matrix modulus function, i.e., the element-wise absolute value as in [170]. In the following, we derive the expressions on the two sides of (4.9) for detection and isolation.

For the adaptive threshold given in the right-hand side of (4.9), we have the known noise bound $\bar{\boldsymbol{n}}$. A bound on the state error $\tilde{\boldsymbol{x}}$, however, is more involved to derive. Following the same approach of [134], we first derive the state error dynamics by subtracting (4.7a) from (4.6a):

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}}) + \boldsymbol{g}(\boldsymbol{u}) - \boldsymbol{g}^H(\boldsymbol{u}) - \boldsymbol{\Lambda}\tilde{\boldsymbol{x}} - \boldsymbol{\Lambda}\boldsymbol{n} + \boldsymbol{d} \tag{4.10}$$

If we further assume healthy conditions, (4.10) takes the form:

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}}) + \boldsymbol{\Lambda}\tilde{\boldsymbol{x}} - \boldsymbol{\Lambda}\boldsymbol{n} + \boldsymbol{d} \tag{4.11}$$

After rearranging terms and integrating both sides of the equation we get:

$$\tilde{\boldsymbol{x}} = e^{-\Lambda t}\tilde{\boldsymbol{x}}(0) + \int_0^t e^{\Lambda(\tau - t)}\left[\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}}) - \Lambda\boldsymbol{n} + \boldsymbol{d}\right] d\tau \tag{4.12}$$

Here we rely on the fact that our system's nonlinear function f is Lipschitz, meaning that any two nearby states produce only a proportionally small difference in f. If we then pick our diagonal observer gains large enough (i.e. larger than that Lipschitz constant), the "correction" term in the observer will always dominate any mismatch caused by the nonlinearity. In other words, the observer error is driven to shrink at an exponential rate, and a finite residual threshold can always be computed. The observer stability can be proven based on Theorem 4.3 in [134]. Nevertheless, this equation cannot be evaluated as $\boldsymbol{n}$ and $\boldsymbol{d}$ are unknown. Nevertheless, we can look for a proper bound of $\tilde{\boldsymbol{x}}$ based on the boundness assumptions for $\boldsymbol{n}$ and $\boldsymbol{d}$. We then have:

$$|\tilde{\boldsymbol{x}}| = \left| e^{-\Lambda t}\tilde{\boldsymbol{x}}(0) + \int_0^t e^{\Lambda(\tau - t)}\left[\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}}) - \Lambda\boldsymbol{n} + \boldsymbol{d}\right] d\tau \right| \tag{4.13}$$

which by leveraging the triangle inequality becomes:

$$\begin{aligned} |\tilde{\boldsymbol{x}}| \leq\ & \underbrace{\left|e^{-\Lambda t}\right| |\tilde{\boldsymbol{x}}(0)|}_{\boldsymbol{\alpha}} \\ & + \underbrace{\int_0^t \left|e^{\Lambda(\tau - t)}\right| \underbrace{\left(|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\hat{\boldsymbol{x}})| + \Lambda\bar{\boldsymbol{n}} + \bar{\boldsymbol{d}}\right)}_{\xi} d\tau}_{\boldsymbol{\beta}} \end{aligned} \tag{4.14}$$

The homogeneous term $\boldsymbol{\alpha}$ depends on the initial state error for which we have $|\tilde{\boldsymbol{x}}(0)| \leq |\boldsymbol{y}(0)| + \bar{\boldsymbol{n}} + \bar{\tilde{\boldsymbol{x}}}$ where $\bar{\tilde{\boldsymbol{x}}}$ is a known bound in the initial state estimate. This term will die out because of the exponential term in a negative power. Term $\boldsymbol{\beta}$ can be computed by numerical integration of $\dot{\boldsymbol{\beta}} = -\boldsymbol{\Lambda}\boldsymbol{\beta} + \boldsymbol{\xi}$ with zero initial conditions. While we can have the state estimate $\hat{\boldsymbol{x}}$ to evaluate $\boldsymbol{f}(\hat{\boldsymbol{x}})$ in (4.14), the state $\boldsymbol{x}$ is unknown and thus the term $\boldsymbol{f}(\boldsymbol{x})$ is unknown as well. Nevertheless, we can substitute $\boldsymbol{x} = \boldsymbol{y} - \boldsymbol{n}$ and then expand the expression $\boldsymbol{f}(\boldsymbol{y} - \boldsymbol{n})$ by leveraging again the triangular inequality to get an upper bound for the right-hand side of (4.14). Thus, a bound for the right-hand side of (4.8) can be computed and will be denoted as $\bar{\boldsymbol{\epsilon}}$ with:

$$|\tilde{\boldsymbol{x}}| + \bar{\boldsymbol{n}} \leq \underbrace{\bar{\boldsymbol{\epsilon}}}_{\text{threshold}} \tag{4.15}$$

Thus, we have the following inequality that holds in healthy conditions where both terms can be evaluated:

$$|\boldsymbol{\epsilon}(\boldsymbol{y}, \hat{\boldsymbol{y}})| \leq \bar{\boldsymbol{\epsilon}}(\boldsymbol{y}, \hat{\boldsymbol{y}}, \bar{\boldsymbol{n}}, \bar{\boldsymbol{d}}) \tag{4.16}$$

The adaptive threshold varies with respect to the estimate of the system while it takes into account the corruption of this signal from worst-case disturbance and noise signals.

4.3.2 Active Fault Diagnosis based on MPC Reconfiguration

The presence of actuator faults is detected by the following set of analytical redundancy relations (ARRs):

$$\mathcal{E}_i \,:\, |\epsilon_i(t)| - \bar{\epsilon}_i(t) \leq 0, i \in \{x, y, \psi, u, v, r\} \tag{4.17}$$

where $|\epsilon_i(t)|$ and $\bar{\epsilon}_i(t)$ are the elements of $|\boldsymbol{\epsilon}(t)|$ and $\bar{\boldsymbol{\epsilon}}(t)$ in (4.16) respectively. Violation of one of these ARRs at any time instance means that the real system is behaving significantly differently with respect to the healthy system model used in the nonlinear observer. Since this discrepancy is not due to measurement noise or disturbances as they have already been accounted for, we can then conclude that a fault has occurred. The first time instant that (4.17) is invalid for at least one $i \in \{x, y, \psi, u, v, r\}$ signifies the time instant of fault detection defined as:

$$t_{D_i} = \min\{t \,:\, |\epsilon_i(t)| - \bar{\epsilon}_i(t) > 0\} \tag{4.18}$$

Until this instant, we assume that either no faults have occurred or there are faults that have not been detected yet. The binary decision for a detected fault is defined as:

$$D(t) = \begin{cases} 0, & t < t_D \\ 1, & t \geq t_D \end{cases} \tag{4.19}$$

with $t_D = \min\{t_{D_i} \,:\, i \in \{x, y, \psi, u, v, r\}\}$. Thus a fault is detected at any time when $D(t) = 1$.

For isolation, we investigate how each one of the LoE faults η_j, $j \in \{l, r, b\}$ affects the system. First, we need to create a binary Fault Signature Matrix (FSMX) as a reference and then a binary decision vector that through comparison with the FSMX will pinpoint the exact location of the fault. For the FSMX, we need to find the effect of each actuator fault on the residuals, that is, how the discrepancy due to the occurring faults denoted as

	η_l	η_r	η_b
$\mathcal{E}_u$	1	1	0
$\mathcal{E}_v$	1	1	1
$\mathcal{E}_r$	1	1	1

Table 4.1: Actuator FSMX $\boldsymbol{F}$

	η_l	η_r	η_b
$\mathcal{E}_u$	1	1	0
$\mathcal{E}_v$	0	1	1
$\mathcal{E}_r$	1	1	1

(a) FSMX $\boldsymbol{F}_l^R$

	η_l	η_r	η_b
$\mathcal{E}_u$	1	1	0
$\mathcal{E}_v$	1	0	1
$\mathcal{E}_r$	1	1	1

(b) FSMX $\boldsymbol{F}_r^R$

Table 4.2: FSMXs after reconfiguration

$\tilde{\boldsymbol{g}}(\boldsymbol{u}) = \boldsymbol{g}(\boldsymbol{u}) - \boldsymbol{g}^H(\boldsymbol{u})$ affects the error dynamics (4.10). We compute the Jacobian of $\tilde{\boldsymbol{g}}(\boldsymbol{u})$ with respect to the vector of faults $\boldsymbol{\eta} = (\eta_l, \eta_r, \eta_b)^\top$ as:

$$\nabla_{\boldsymbol{\eta}} \tilde{\boldsymbol{g}}(\boldsymbol{u}) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \tilde{\boldsymbol{g}}_{\boldsymbol{ul}}(\tau_l, \alpha_l) & \tilde{\boldsymbol{g}}_{\boldsymbol{ur}}(\tau_r, \alpha_r) & 0 \\ \tilde{\boldsymbol{g}}_{\boldsymbol{vl}}(\tau_l, \alpha_l) & \tilde{\boldsymbol{g}}_{\boldsymbol{vr}}(\tau_r, \alpha_r) & \tilde{\boldsymbol{g}}_{\boldsymbol{vb}}(\tau_b, \alpha_b) \\ \tilde{\boldsymbol{g}}_{\boldsymbol{rl}}(\tau_l, \alpha_l) & \tilde{\boldsymbol{g}}_{\boldsymbol{rr}}(\tau_r, \alpha_r) & \tilde{\boldsymbol{g}}_{\boldsymbol{rb}}(\tau_b, \alpha_b) \end{bmatrix} \tag{4.20}$$

The Jacobian matrix, commonly used in sensitivity analysis, captures the rate of change of the system's output (ARRs in our case) concerning small changes in the actuator faults. Each row of this matrix corresponds to a specific ARR $\mathcal{E}_i$, $i \in \{x, y, \psi, u, v, r\}$, and each column corresponds to a particular actuator fault η_j, $j \in \{l, r, b\}$, offering insights into the impact of each fault on the ARRs. After computing the Jacobian matrix (4.20), it is observed that the first three ARRs ($\mathcal{E}_i$, $i \in \{x, y, \psi\}$) do not exhibit sensitivity to the actuator faults (η_j, $j \in \{l, r, b\}$). Consequently, these states are omitted from the FSMX, as their inclusion would not contribute valuable information regarding fault isolation. The FSMX matrix, denoted as $\boldsymbol{F}$, is constructed based on the relevant ARRs ($\mathcal{E}_i$ where $i \in \{u, v, r\}$), resulting in a focused representation that captures the impact of actuator faults on the observable system dynamics. The matrix is shown in Table 4.1.

From Table 4.1, we can deduce that due to the geometrical symmetry of the actuators, they affect the dynamics in the same way. Geometrical symmetry in this context refers to the similar spatial arrangement or characteristics of the actuators. Specifically, the actuator faults η_l and η_r exhibit identical impacts on the ARRs of the system and thus, distinguishing between these faults becomes challenging. By utilizing overactuation, the key idea in this work is to actively change the input vector $\boldsymbol{u}$ so that the effect of some actuators is nullified in specific ARRs and thus, the faults can be isolable. Furthermore, this needs to be realized while keeping the ASV controllable and able to perform the collision avoidance maneuver needed. Indeed, if we examine matrix $\boldsymbol{F}$ in Table 4.1 we see that the two columns that correspond to faults η_l and η_r are identical. However, by placing a zero in either column at

the row that corresponds to ARR, $\mathcal{E}_v$, all three columns become linearly independent, as shown in Table 4.2. This can be realized by setting the corresponding term of (4.20) to zero, meaning that we want to make that component invariant of the corresponding control action. Solving either $\tilde{\boldsymbol{g}}_{\boldsymbol{vl}}(\tau_l, \alpha_l) = 0$ for α_l or $\tilde{\boldsymbol{g}}_{\boldsymbol{vr}}(\tau_r, \alpha_r) = 0$ for α_r, we get expressions for the azimuths of the form:

$$\alpha_l = \alpha_l^0(\tilde{\boldsymbol{M}}, l_{lr}, w_{lr}), \quad \alpha_r = \alpha_r^0(\tilde{\boldsymbol{M}}, l_{lr}, w_{lr}), \tag{4.21}$$

and since these parameters are constant, α_l and α_r can be set to the constant values α_l^0 and α_r^0 respectively so that they do not affect the sway dynamics and the corresponding ARR, that is, either $\tilde{\boldsymbol{g}}_{\boldsymbol{vl}}(\tau_l, \alpha_l^0) = 0$ or $\tilde{\boldsymbol{g}}_{\boldsymbol{vr}}(\tau_r, \alpha_r^0) = 0$. Choosing for example to nullify the effect of the right azimuth thruster will result in a different FSMX matrix denoted as $\boldsymbol{F}_{\boldsymbol{r}}^{\boldsymbol{R}}$ and shown in Table 4.2b. Note that fulfillment of either equation in (4.21) does not nullify the rest of the terms in (4.20). To reconfigure the actuators for isolation purposes, we leverage the MPC controller in [166] (see Fig. 3.1). The MPC recursively optimizes a multi-objective cost that includes a penalty on the control inputs accounting at the same time for collision avoidance and system constraints. The part of the cost function of our MPC controller (relevant for the reconfiguration here) regarding the right azimuth input takes the form:

$$J_{\alpha_r}(\boldsymbol{u}_k) = q_{\alpha_r}\alpha_r^2 + q_{\alpha_r^0}(\alpha_r - \alpha_r^0)^2 \tag{4.22}$$

where q_{α_r} and $q_{\alpha_r^0}$ are tuning penalty weights and their value depends on whether we are in normal conditions or the reconfiguration mode is activated. More specifically, a higher value of $q_{\alpha_r^0}$ *forces* the optimizer to stir $\alpha_r \to \alpha_r^0$. If the actuator is faulty, then, the real system will not be able to follow the commanded action. If it is healthy, we are able to isolate the fault on the other actuator. To complete the isolation, we need to derive the binary decision vector to be compared with the updated FSMX $\boldsymbol{F}^{\boldsymbol{r}}$. The binary decision vector is obtained as:

$$\boldsymbol{D} = (D_u, D_v, D_r)^\top \tag{4.23}$$

with:

$$D_i(t) = \begin{cases} 0, & t < t_{D_i} \\ 1, & t \geq t_{D_i} \end{cases} \tag{4.24}$$

with $t_{D_i} : i \in \{u, v, r\}$ the time that the i^{th} ARR is violated for the first time. The active FDI logic is summarized in Algorithm 2.

Algorithm 2 Proposed Fault Diagnosis Logic

Input: $\boldsymbol{y}(t)$, $\boldsymbol{u}(t)$
Output: Fault ID: $\{r, l, b\}$

for $t = 1, 2, \ldots$ **do**
  Compute $|\boldsymbol{\epsilon}(\boldsymbol{y}(t), \hat{\boldsymbol{y}}(t))|$
  Compute $\bar{\boldsymbol{\epsilon}}(\boldsymbol{y}, \hat{\boldsymbol{y}}, \bar{\boldsymbol{n}}, \bar{\boldsymbol{d}})$
  Compute $D(t)$
  if $D(t) = 1$ **then**
    Compute $\boldsymbol{D}(t)$
    if $\boldsymbol{D}(t) = \boldsymbol{F}(:, 3)$ **then**
      Fault ID: b $(\eta_b \neq 1)$
    else
      $q_{\alpha_r} \leftarrow 0$
      $q_{\alpha_r^0} \leftarrow 10^6$
      Compute $\boldsymbol{D}(t)$ from Eq. (4.23)
      if $\boldsymbol{D}(t) = \boldsymbol{F}(:, 2)$ **then**
        Fault ID: r $(\eta_r \neq 1)$
      else $\boldsymbol{D}(t) = \boldsymbol{F}(:, 1)$
        Fault ID: l $(\eta_l \neq 1)$
      end if
    end if
  end if
end for

4.4 Results

This section presents simulation results to validate the efficacy of our algorithm in a simple traffic scenario. Our framework is implemented in ROS: the controller and FDI module in C++ and the simulator of the ASV and OV in Python. The algorithm runs in an Ubuntu machine with an Intel i7 CPU@1.8GHz and 16GB of RAM.

In this simple traffic scenario, the ASV is obliged by the traffic rules to avoid collision by turning to its starboard (right side) and passing behind the OV. While the ASV is performing the collision avoidance maneuver, at time $t_F = 7s$ we inject a permanent fault to the right thruster, $\eta_r = 0.2$. In Figure 4.3 we can see the instances of the ASV at the time the fault occurs $t_F = 7$ sec, the fault is detected $t_D = 7.86s$, and lastly the fault is isolated $t_I = 14.15s$. Notice that the time between fault occurrence t_F and isolation t_I is relatively small, i.e., the ASV has not traversed a large distance, highlighting the aptness of the diagnosis procedure. Figures 4.4, 4.5, and 4.6 show the norm of the residual coupled with its adaptive threshold and the violation decision for each one of the three ARRs related to the velocity states that contribute to the diagnosis procedure. The first violation is noticed in Figure 4.4 at $t_D = 7.86s$. Reconfiguration starts subsequently to nullify the effect of the faulty thruster on $\mathcal{E}_v$ as seen in Figure 4.5. After a few seconds, at time $t_I = 14.15s$ seconds, a violation of $\mathcal{E}_r$ indicates that the fault has occurred in the right thruster, and thus, isolation is completed. Lastly, Figure 4.7 shows the control input in blue solid lines along with the faulty input signal $\tau_r^F = \eta_r \tau_r$ that is applied on the ASV plotted in a light blue dashed line right after the fault has occurred. In the same figure, the reconfigured control input α_r is plotted in red to show the constant value it has been imposed for isolation purposes.

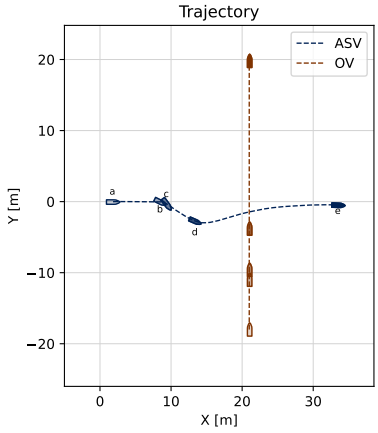

Figure 4.3: Plotted trajectories with instances of the vessels at a) initial time, b) time of fault occurrence, c) time of detection, d) time of isolation, and e) final time.

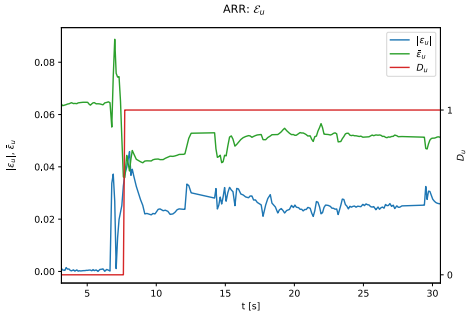

Figure 4.4: The residual norm, threshold, and decision for the ARR corresponding to the surge velocity state u.

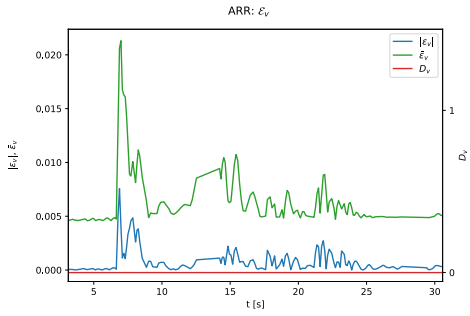

Figure 4.5: The residual norm, threshold, and decision for the ARR corresponding to the sway velocity state v.

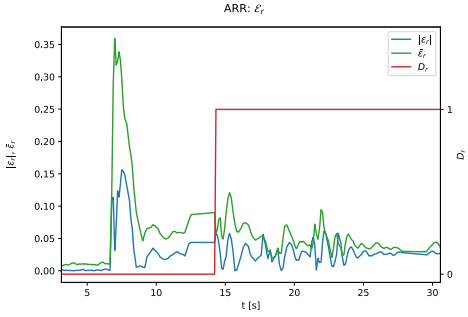

Figure 4.6: The residual norm, threshold, and decision for the ARR corresponding to the yaw velocity state r.

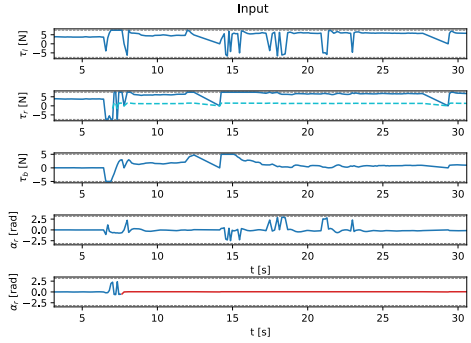

Figure 4.7: Input signals with the actual input signal τ_r (light blue dashed line) after the fault has occurred and the input signal α_r (red line) after reconfiguration.

4.5 Conclusions

This chapter proposed a thruster fault diagnosis algorithm for an ASV, comprised of a cascaded interconnection of a detection and an isolation module as a response to Research Question **Q2**: "How to detect and isolate actuator faults in ASVs to enhance overall operational safety and reliability?". For detection, residuals generated by nonlinear observers are coupled with adaptive thresholds that accommodate noise and disturbance bounds to enhance robustness against false alarms. For isolation, we rely on the system's redundancy in actuation and the capability to set actuation constraints in our MPC controller so that we can isolate thruster faults that otherwise would be indistinctive due to the system's symmetry. Simulation results demonstrate the effectiveness of this methodology. In the following chapter, we explore a method to estimate the fault parameters along with a feasible parameter set in order to get a robust estimation. This will be the cornerstone to extend our method with reconfiguration and influence the planning process, contributing to safer and more adaptive collision avoidance maneuvers.

5

Set-Membership Estimation for Fault Diagnosis

*This chapter introduces a Fault Diagnosis (Detection, Isolation, and Estimation) method using Set-Membership Estimation (SME) designed for a class of nonlinear systems that are linear to the fault parameters as a response to Research Question **Q3**: "How can fault parameters be accurately and robustly estimated under varying operational conditions, including the presence of disturbances and noise?" The methodology advances fault diagnosis by continuously evaluating an estimate of the fault parameter and a feasible parameter set where the true fault parameter belongs. Unlike previous SME approaches, in this work, we address nonlinear systems subjected to both input and output uncertainties by utilizing inclusion functions and interval arithmetic. Additionally, we present an approach to outer-approximate the polytopic description of the feasible parameter set by effectively balancing approximation accuracy with computational efficiency resulting in improved fault detectability. Lastly, we introduce adaptive regularization of the parameter estimates to enhance the estimation process when the input-output data are sparse or non-informative, enhancing fault identifiability. We demonstrate the effectiveness of this method in simulations involving an Autonomous Surface Vehicle in both a path-following and a realistic collision avoidance scenario, underscoring its potential to enhance safety and reliability in critical applications. This chapter is structured as follows: Section 5.1 introduces the topic of this chapter. Section 5.2 sets the problem formulation. Sections 5.4 to 5.7 detail our FD method within the SME framework. Section 5.8 demonstrates the efficacy of the method through both a simple path-following and a collision avoidance scenario. We conclude with final remarks in Section 5.9.*

This chapter is based on **A. Tsolakis**, *L. Ferranti, and V. Reppa, "Set-Membership Estimation for Fault Diagnosis of Nonlinear Systems," submitted to European Control Conference, Oct. 2024, to be presented in June 2025.*

5.1 Introduction

As autonomous systems evolve, they increasingly rely on sophisticated technology and complex hardware, raising significant challenges in ensuring safe and reliable operation. Critical components such as sensors, actuators, and computational units present safety and reliability risks, as faults in these components can lead to potentially catastrophic failures. To address this, robust mechanisms for detecting and mitigating faults are essential, particularly those applicable across a range of mobile robotic platforms, including ground, marine, and aerial systems.

In the previous chapter, we introduced an FDIE method based on residuals coupled with adaptive thresholds, along with an isolation procedure to detect and localize actuator faults. While this approach guarantees no false alarms and has proven effective, it operates in the state space, offering no direct information about the fault parameter's value. That is, while we know *if* something is wrong and *where*, we lack insight into *how much*. Fault isolation in this work, however, required system redundancy to distinguish between different faults to ensure that the diagnosis process remains accurate despite uncertainties.

An alternative approach that addresses these limitations and operates directly in the parameter space is SME. SME is widely used in (FDIE), providing a direct method by employing inverse tests to detect faults while estimating the set of feasible fault parameters based on past input-output data. The process begins by computing the Unfalsified Parameter Set (UPS), which represents the set of fault parameter values consistent with the system's evolution given the bounded uncertainties. Over time, the intersection of successive UPS results in the Feasible Parameter Set (FPS), which contains fault parameters still consistent with past measurements of the system's evolution. Fault detection occurs when the FPS becomes empty, indicating that no parameter values remain consistent with the system's measurements. Fault isolation is realized similarly, after properly projecting the FPS in the different directions in the parameter space but without the need for system redundancy. Concurrently, a nominal parameter estimate is computed online, offering a direct quantification of the fault parameter.

Recent work has renewed interest in SME, particularly in adaptive control. Contributions such as [171] have combined SME with Model Predictive Control (MPC) in a Robust Adaptive MPC (RAMPC) framework, enabling planning based on nominal parameters while maintaining robustness against all feasible parameter realizations. Extensions of SME in RAMPC frameworks for linear systems have been explored in [172, 173, 174], and for nonlinear systems in [175], though the challenge of handling both state and output uncertainties simultaneously remains largely unaddressed.

Inspired by SME's suitability for FD and its compatibility with MPC in a RAMPC framework, this work proposes an FD method based on SME, aimed at enhancing the trajectory optimization method introduced in [166] to improve safety in environments shared with human-operated vehicles. Specifically, this work extends SME to *nonlinear systems* affected by both *state disturbances* and *measurement noise*—a gap in the current state of the art. The method employs an inverse test for fault detection and isolation, with fault estimation achieved through continuous updates to the feasible parameter set and a fault parameter estimate. The key contributions of this work are:

- Set-membership estimation to nonlinear systems, accounting for both disturbances and measurement noise. This capability ensures *false alarm immunity* by design,

thereby increasing the robustness of the fault detection process.

- A tighter outer approximation of the feasible parameter set that balances accuracy and computational efficiency, based on user-defined preferences. This leads to improved *fault detectability*, reducing the risk of missed detections and enhancing the system's responsiveness to faults.

- Adaptive regularization in fault parameter estimation to handle cases of sparse, non-informative measurement data, resulting in improved *fault identifiability*.

5.2 Problem Formulation

Consider the following discrete, nonlinear system, which is linear in the vector of fault parameters denoted as $\boldsymbol{\theta} \in \mathbb{R}^p$:

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t) + G(\boldsymbol{u}_t)\boldsymbol{\theta} + \boldsymbol{d}_t \tag{5.1}$$

where $\boldsymbol{x}_t \in \mathbb{R}^n$ is the state, $\boldsymbol{u}_t \in \mathbb{R}^m$ is the input, and $\boldsymbol{d}_t \in \mathbb{R}^n$ is the unknown disturbance acting on the system. We assume that the autonomous map $\boldsymbol{f}(\cdot) \in \mathbb{R}^n$ and the input map $G(\cdot) \in \mathbb{R}^{n\times p}$ are both known. Additionally, we assume that the full state of the system can be measured, albeit corrupted by measurement noise, as:

$$\boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{n}_t \tag{5.2}$$

where $\boldsymbol{y}_t \in \mathbb{R}^n$ is the state measurement, and $\boldsymbol{n}_t \in \mathbb{R}^n$ is the unknown additive measurement noise.

Assumption 1 The disturbance $\boldsymbol{d}_t$ and noise $\boldsymbol{n}_t$ are unknown but bounded signals with known bounds denoted as $\bar{\boldsymbol{d}}$ and $\bar{\boldsymbol{n}}$, respectively:

$$|\boldsymbol{d}_t| \leq \bar{\boldsymbol{d}}, \quad \forall t = 1, 2, \ldots \tag{5.3}$$

$$|\boldsymbol{n}_t| \leq \bar{\boldsymbol{n}}, \quad \forall t = 1, 2, \ldots \tag{5.4}$$

The inequalities between vectors are to be interpreted element-wise, where $|\cdot|$ denotes the matrix modulus function, i.e., the element-wise absolute value.

Assumption 2 The fault parameter vector $\boldsymbol{\theta} \in \mathbb{R}^p$ is time-invariant, with elements $\theta_i \in [0, 1]$, $i = 1, 2, \ldots, p$, describing the health of the system. The values of θ_i represent the following conditions:

$$\theta_i = \begin{cases} \theta_i = 1, \ \forall i \in \{1, 2, \ldots, p\}, & \text{healthy system} \\ \theta_i < 1, \ \exists i \in \{1, 2, \ldots, p\}, & \text{faulty system} \end{cases} \tag{5.5}$$

The objective of this work is to detect, isolate, and estimate faults with guarantees, using SME to compute an outer approximation of the feasible parameter set and an estimate of the fault parameter. Inverse tests are then applied to detect faults, isolate them, and refine fault parameter estimates upon detection.

5.3 Method Overview

The steps involved in SME are illustrated in Figure 5.1. First, Section 5.4 describes the computation of the Unfalsified Parameter Set (UPS), denoted as $\mathbf{\Delta}_t \subseteq \mathbb{R}^p$, at each time step t, based on the latest input-output measurements (top-left). Subsequently, in Section 5.5, the Feasible Parameter Set (FPS), denoted as $\mathbf{\Theta}_t \subseteq \mathbb{R}^p$, is recursively computed based on the intersection of the previous FPS, $\mathbf{\Theta}_{t-1}$, and the current update from $\mathbf{\Delta}_t$ (top-right). The FPS is then outer-approximated by a simpler polytope, denoted as $\overline{\mathbf{\Theta}}_t \subseteq \mathbb{R}^p$, which is described by a predefined number of maximum directions computed offline according to the required trade-off between accuracy and efficiency (bottom-right). In Section 5.6, an estimate $\hat{\boldsymbol{\theta}}_t \in \overline{\mathbf{\Theta}}_t$ is derived, accounting for the quality of the available measurements by incorporating adaptive regularization (bottom-left). Finally, Section 5.7 describes the FDE method, which relies on the components computed in the preceding sections.

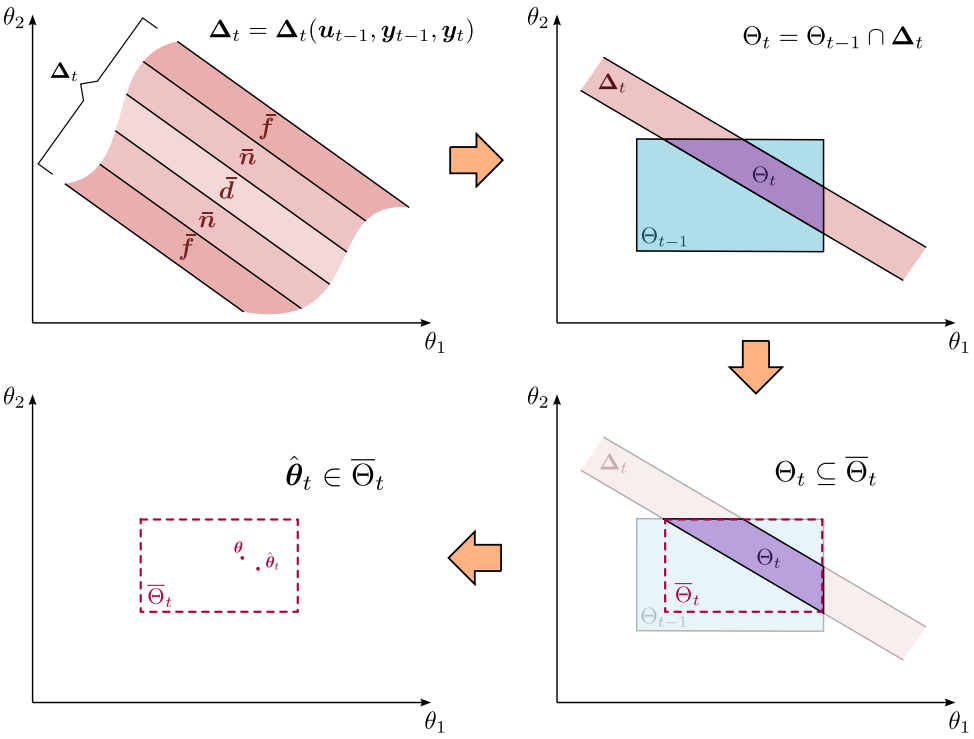

Figure 5.1: Overview of the different steps in SME in a 2-D example: First, the UPS, $\mathbf{\Delta}_t$, is computed at each time step t, based on the latest input-output measurements. The FPS, denoted as $\mathbf{\Theta}_t$, is recursively computed based on the intersection of the existing FPS, $\mathbf{\Theta}_{t-1}$, and the current update from $\mathbf{\Delta}_t$. The FPS is then outer-approximated by a simpler polytope, denoted as $\overline{\mathbf{\Theta}}_t$. Lastly, an estimate, denoted as $\hat{\boldsymbol{\theta}}_t \in \overline{\mathbf{\Theta}}_t$, is computed.

5.4 Unfalsified Parameter Set

In this section, we introduce a method to compute the Unfalsified Parameter Set (UPS) based on input-output measurements. First, we express the disturbance and noise bounds

in polytopic form:

$$|\boldsymbol{d}_t| \leq \bar{\boldsymbol{d}} \Leftrightarrow \boldsymbol{d}_t \in \mathcal{D} = \{\boldsymbol{d}_t \in \mathbb{R}^n | \boldsymbol{H}\boldsymbol{d}_t \leq \boldsymbol{h}_{\boldsymbol{d}}\} \tag{5.6}$$

$$|\boldsymbol{n}_t| \leq \bar{\boldsymbol{n}} \Leftrightarrow \boldsymbol{n}_t \in \mathcal{N} = \{\boldsymbol{n}_t \in \mathbb{R}^n | \boldsymbol{H}\boldsymbol{n}_t \leq \boldsymbol{h}_{\boldsymbol{n}}\} \tag{5.7}$$

where $\boldsymbol{H} = \begin{bmatrix} \boldsymbol{I}_n & -\boldsymbol{I}_n \end{bmatrix}^\top \in \mathbb{R}^{2n \times n}$ with $\boldsymbol{I}_n \in \mathbb{R}^{n \times n}$ the identity matrix of dimension n, $\boldsymbol{h}_{\boldsymbol{d}} = \begin{bmatrix} \bar{\boldsymbol{d}} & \bar{\boldsymbol{d}} \end{bmatrix}^\top \in \mathbb{R}^{2n}$ and $\boldsymbol{h}_{\boldsymbol{n}} = \begin{bmatrix} \bar{\boldsymbol{n}} & \bar{\boldsymbol{n}} \end{bmatrix}^\top \in \mathbb{R}^{2n}$. Combining equations (5.1) and (5.2) yields:

$$\boldsymbol{y}_{t+1} - \boldsymbol{G}(\boldsymbol{u}_t)\boldsymbol{\theta} = \boldsymbol{d}_t + \boldsymbol{n}_{t+1} + \boldsymbol{f}(\boldsymbol{x}_t) \tag{5.8}$$

where $\boldsymbol{u}_t$ and $\boldsymbol{y}_t$ are known signals (kept on the left-hand side) while $\boldsymbol{d}_t$ and $\boldsymbol{n}_t$ are unknown but bounded based on Assumption 1. The complication arises with the remaining term $\boldsymbol{f}(\boldsymbol{x}_t)$ which depends on both the known state measurement $\boldsymbol{y}_t$ and the unknown noise signal $\boldsymbol{n}_t$ since $\boldsymbol{x}_t = \boldsymbol{y}_t - \boldsymbol{n}_t$. To handle this term, we will rely on interval analysis [176] to compute lower and upper bounds for $\boldsymbol{f}(\cdot)$. To do this, we first need to find an interval for the state $\boldsymbol{x}_t$ from (5.2) and (5.4) as follows:

$$\underline{\boldsymbol{x}}_t = \boldsymbol{y}_t - \bar{\boldsymbol{n}} \leq \boldsymbol{x}_t \leq \boldsymbol{y}_t + \bar{\boldsymbol{n}} = \overline{\boldsymbol{x}}_t \Leftrightarrow \boldsymbol{x}_t \in [\underline{\boldsymbol{x}}_t, \overline{\boldsymbol{x}}_t] = [\boldsymbol{x}_t] \tag{5.9}$$

which is time-varying and can be updated online with each new state measurement $\boldsymbol{y}_t$. Usually, an inclusion function $\mathbb{f}(\cdot)$ is found for $\boldsymbol{f}(\cdot)$ based on the state interval (5.9) satisfying:

$$\boldsymbol{f}([\boldsymbol{x}]) \subset \mathbb{f}(\cdot) \tag{5.10}$$

where $\boldsymbol{f}([\boldsymbol{x}])$ denotes the minimal inclusion function. Computing $\boldsymbol{f}([\boldsymbol{x}])$ would give the tightest possible bounds, but this requires solving two global, non-convex optimization problems which are prohibitive to solve online. A schematic example of an inclusion function and the minimal inclusion function is illustrated in Figure 5.2. Instead, we can use

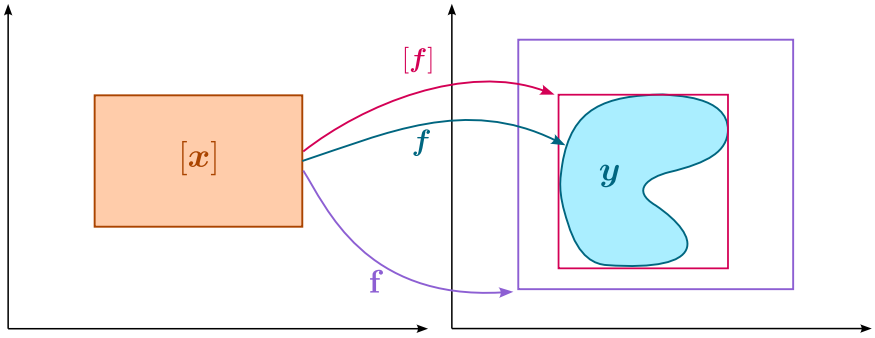

Figure 5.2: Schematic representation of an inclusion function and the minimal inclusion function that tightly bounds the function for a given state interval.

interval arithmetic which extends math operations and elementary functions to intervals and is much more computationally efficient. Since the expression of $\boldsymbol{f}(\cdot)$ is known, we can compute an interval for the term as:

$$[\underline{\boldsymbol{f}}_t, \overline{\boldsymbol{f}}_t] = \boldsymbol{I}([\underline{\boldsymbol{x}}_t, \overline{\boldsymbol{x}}_t]) \supseteq \boldsymbol{f}([\boldsymbol{x}_t]) \tag{5.11}$$

where $\boldsymbol{I}(\cdot)$ denotes an appropriate function to compute intervals via interval arithmetic. In practice, we use C++ BOOST Interval Arithmetic library [177] which can efficiently compute such intervals online. Note that the interval $[\underline{\boldsymbol{f}}_t, \overline{\boldsymbol{f}}_t]$ is also time-varying as it implicitly depends on the state measurement $\boldsymbol{y}_t$. With these known bounds for the system dynamics $\boldsymbol{f}(\cdot)$ we can formulate similar polytopic bounds for the autonomous term in (5.8) similar to those derived before:

$$\begin{aligned} &\underline{\boldsymbol{f}}_t \leq \boldsymbol{f}(\boldsymbol{x}_t) \leq \overline{\boldsymbol{f}}_t \Leftrightarrow \\ &\qquad \boldsymbol{f}(\boldsymbol{x}_t) \in \mathcal{F}_t = \{\boldsymbol{f}(\boldsymbol{x}_t) \in \mathbb{R}^n \,|\, \boldsymbol{H}\boldsymbol{f}(\boldsymbol{x}_t) \leq \boldsymbol{h}_f(\boldsymbol{y}_t)\} \end{aligned} \tag{5.12}$$

with $\boldsymbol{h}_f(\boldsymbol{y}_t) = \begin{bmatrix} \overline{\boldsymbol{f}}_t & -\underline{\boldsymbol{f}}_t \end{bmatrix}^\top \in \mathbb{R}^{2n}$. The key observation is that the polytopic inequalities in (5.6), (5.7) and (5.12) are constructed such that all the unknown signals are multiplied from the left with the same matrix $\boldsymbol{H}$. Additionally, from (5.4) we know that $|\boldsymbol{n}_t| \leq \bar{\boldsymbol{n}}, \forall t$ and thus from (5.7) we can also deduce that $\boldsymbol{H}\boldsymbol{n}_{t+1} \leq \boldsymbol{h}_{\boldsymbol{n}}$. We can then sum these inequalities and factor out $\boldsymbol{H}$ to get:

$$\boldsymbol{H}(\boldsymbol{d}_t + \boldsymbol{n}_{t+1} + \boldsymbol{f}(\boldsymbol{x}_t)) \leq \boldsymbol{h}_{\boldsymbol{d}} + \boldsymbol{h}_{\boldsymbol{n}} + \boldsymbol{h}_f(\boldsymbol{y}_t) \tag{5.13}$$

Notice that the right-hand side term in (5.8) appears in the left-hand side of (5.13). After substituting (5.8) to (5.13) and rearranging the terms we get:

$$-\boldsymbol{H}\boldsymbol{G}(\boldsymbol{u}_t)\boldsymbol{\theta} \leq \boldsymbol{h}_{\boldsymbol{d}} + \boldsymbol{h}_{\boldsymbol{n}} + \boldsymbol{h}_f(\boldsymbol{y}_t) - \boldsymbol{H}\boldsymbol{y}_{t+1} \tag{5.14}$$

If we now shift the timestep one step backward, we can derive the UPS as:

$$\Delta_t = \{\boldsymbol{\theta} \in \mathbb{R}^p \mid -\boldsymbol{H}\boldsymbol{G}(\boldsymbol{u}_{t-1})\boldsymbol{\theta} \leq \boldsymbol{h}_{\boldsymbol{d}} + \boldsymbol{h}_{\boldsymbol{n}} + \boldsymbol{h}_f(\boldsymbol{y}_{t-1}) - \boldsymbol{H}\boldsymbol{y}_t\} \tag{5.15}$$

which can be computed at each time step t based on the input-output measurement set $\{\boldsymbol{u}_{t-1}, \boldsymbol{y}_{t-1}, \boldsymbol{y}_t\}$. Notice that (5.15) is a similar expression to the ones found in [173] and [174] with the difference that here we have additional bounding terms on the right-hand side of the inequality in (5.15) to account for measurement noise: Term $\boldsymbol{h}_{\boldsymbol{n}}(\bar{\boldsymbol{n}})$ directly sums the measurement bounds while $\boldsymbol{h}_f(\boldsymbol{y}_{t-1}, \bar{\boldsymbol{n}})$ is a time-varying term that sums the noise bounds implicitly after they are mapped through the nonlinear autonomous term $\boldsymbol{f}(\boldsymbol{x}_t) = \boldsymbol{f}(\boldsymbol{y}_t, \boldsymbol{n}_t)$ via interval arithmetic. This is illustrated schematically in Figure 5.3 in comparison with not accounting for the measurement noise. The UPS is therefore dilated due to the measurement noise as in [174] but its dilation is generalized here to nonlinear systems that are linear to the parameters by employing interval analysis. Notice that in this formulation, even if the system is nonlinear, the assumption that the system is linear to the parameters maintains the same polytopic description for the UPS as a p-dimensional slab in the parameter space which is advantageous for the computation of the FPS, $\boldsymbol{\Theta}_t$, as explained in the next section.

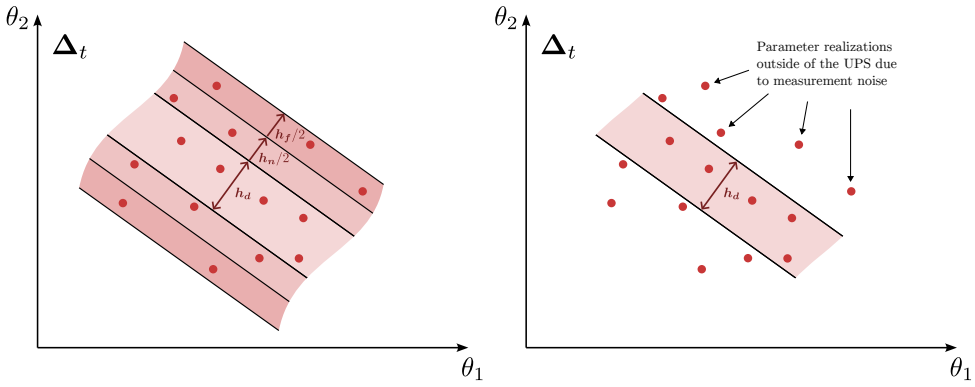

Figure 5.3: On the right is our formulation of the UPS that additionally accounts for measurement noise and thus all possible parameter realizations are guaranteed to lie inside the UPS. In contrast, if the noise is not accounted for, several realizations will be outside the UPS.

5.5 Feasible Parameter Set

In order to compute the Feasible Parameter Set (FPS), Θ_t, we begin with expressing some initial parameter bounds in polytopic form:

$$\boldsymbol{\theta} \in \Theta_0 = \{\boldsymbol{\theta} \in \mathbb{R}^p | \boldsymbol{H}_{\theta_0} \boldsymbol{\theta} \leq \boldsymbol{h}_{\theta_0}\} \tag{5.16}$$

where $\boldsymbol{H}_{\theta_0} = \begin{bmatrix} \boldsymbol{I}_p & -\boldsymbol{I}_p \end{bmatrix}^\top \in \mathbb{R}^{2p \times p}$ with $\boldsymbol{I}_p \in \mathbb{R}^{p \times p}$ the identity matrix of dimension p, and $\boldsymbol{h}_{\theta_0} = \begin{bmatrix} \underline{\theta} & \overline{\theta} \end{bmatrix}^\top$ has known, lower and upper parameter bounds. The parameter set Θ_0 is defined as the initial p-dimensional hypercube which describes the range of values of the parameters of interest. The FPS is recursively updated using the UPS, Δ_t, from (5.15) at each time step starting from the initial FPS, Θ_0, as:

$$\Theta_t = \Theta_{t-1} \cap \Delta_t, \quad \forall t = 1, 2, ..., N \tag{5.17}$$

If we describe the FPS in polytopic form:

$$\Theta_t = \{\boldsymbol{\theta} \in \mathbb{R}^p | \boldsymbol{H}_{\theta_t} \boldsymbol{\theta} \leq \boldsymbol{h}_{\theta_t}\} \tag{5.18}$$

and rewrite (5.15) with a simplified notation as:

$$\Delta_t = \{\boldsymbol{\theta} \in \mathbb{R}^p \mid \boldsymbol{H}_{\Delta_t} \boldsymbol{\theta} \leq \boldsymbol{h}_{\Delta_t}\} \tag{5.19}$$

with $\boldsymbol{H}_{\Delta_t} = -\boldsymbol{H}\boldsymbol{G}(\boldsymbol{u}_{t-1})$ and $\boldsymbol{h}_{\Delta_t} = \boldsymbol{h_d} + \boldsymbol{h_n} + \boldsymbol{h}_f(\boldsymbol{y}_{t-1}) - \boldsymbol{H}\boldsymbol{y}_t$ then the intersection of sets described in (5.17) is the concatenation of the inequalities that characterize the FPS in (5.18) and the UPS in (5.19):

$$\boldsymbol{H}_{\theta_t} = \begin{bmatrix} \boldsymbol{H}_{\theta_{t-1}} \\ \boldsymbol{H}_{\Delta_t} \end{bmatrix}, \quad \boldsymbol{h}_{\theta_t} = \begin{bmatrix} \boldsymbol{h}_{\theta_{t-1}} \\ \boldsymbol{h}_{\Delta_t} \end{bmatrix} \tag{5.20}$$

This process works on the condition that the latest set of measurements is informative enough to ensure that the newly introduced inequalities are not redundant. However, continuously concatenating the inequalities, expressed as $\boldsymbol{H}_{\theta_t} = \begin{bmatrix} \boldsymbol{H}_{\theta_{t-N}} & ... & \boldsymbol{H}_{\theta_{t-1}} & \boldsymbol{H}_{\theta_t} \end{bmatrix}^\top \in$

$\mathbb{R}^{2Np\times p}$ and $\boldsymbol{h}_{\boldsymbol{\theta}_t} = \begin{bmatrix}\boldsymbol{h}_{\boldsymbol{\theta}_{t-N}} & \dots & \boldsymbol{h}_{\boldsymbol{\theta}_{t-1}} & \boldsymbol{h}_{\boldsymbol{\theta}_t}\end{bmatrix}^\top \in \mathbb{R}^{2Np}$ quickly becomes impractical since the size of $\boldsymbol{H}_{\boldsymbol{\theta}_t}$ and $\boldsymbol{h}_{\boldsymbol{\theta}_t}$ grows unbounded as $N \to \infty$. In [173] this is handled by outer-approximating the FPS with hypercubes encompassing the derived polytope at each timestep t. In [172] this is generalized by using predefined normal directions of the facets of a polytope that bounds the estimated parameter set and then solving an optimization problem to "tighten" the polytope around the FPS. We propose a method that can efficiently provide an outer bound of the FPS with the accuracy of the bound adjustable depending on the balance between required precision and available computational resources. The computation of the FPS is illustrated in Figure 5.4 and consists of the following steps:

1. Recursively compute a set of predefined, normalized directions, $\mathcal{E} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, ...\}$, normal to the faces of the outer-approximating polytope. These directions are based on the number of parameters p and a user-defined accuracy iterator ϕ from Algorithm 3. This is computed offline.
2. Compute the new FPS $\boldsymbol{\Theta}_t = \overline{\boldsymbol{\Theta}}_{t-1} \cap \boldsymbol{\Delta}_t$ based on the outer-approximation of the previous time step and the new UPS computed at time t starting with $\overline{\boldsymbol{\Theta}}_0 = \boldsymbol{\Theta}_0$.
3. Compute the set of vertices $\mathcal{V}_t = \{\boldsymbol{v}_t^1, \boldsymbol{v}_t^2, ...\}$ of the convex polytope $\boldsymbol{\Theta}_t$.
4. Compute the outer approximation $\overline{\boldsymbol{\Theta}}_t$ of the convex polytope $\boldsymbol{\Theta}_t$ based on the set of vertices $\mathcal{V}_t$ and the set of predefined directions $\mathcal{E}$ from Algorithm 4.
5. Go back to Step 2.

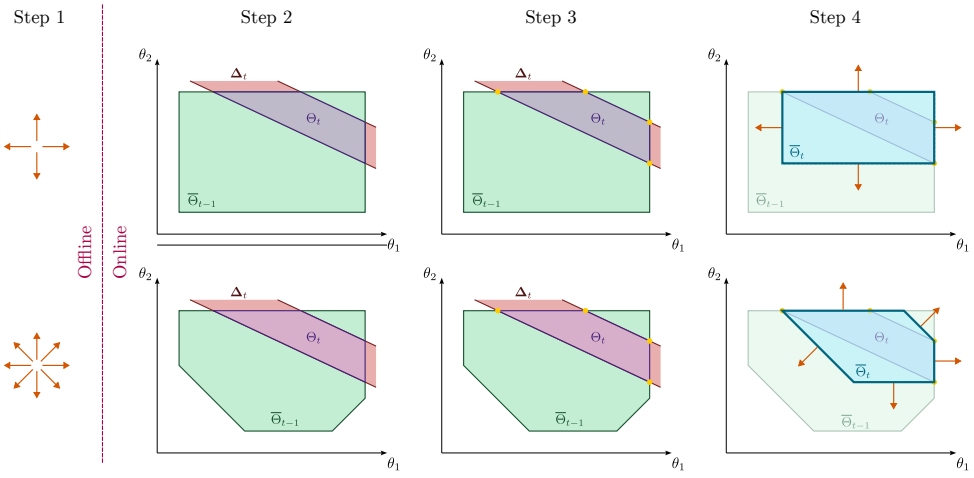

Figure 5.4: The steps to compute the FPS. This starts with computing the predefined directions offline (Step 1). Then the process continues with computing the new FPS from the current UPS (Step 2), computing the vertices of the new FPS (Step 3), and lastly outer-approximating the new FPS based on the predefined directions (Step 4). The process starts again from Step 2, starting from the new outer approximation. Different choices of predefined directions will result in a different outer approximation of the FPS (top and bottom rows).

We propose an algorithm that systematically generates predefined directions for the faces of the approximation polytope offline, with arbitrarily high complexity, and for any number of

parameters p. The generation of predefined directions $\mathcal{E}$ begins with a simple p-dimensional hypercube. The key idea is to "bisect" each edge formed by the intersection of adjacent faces and create a new face with a normal vector that points symmetrically between the normal directions of the intersecting faces. This process can be repeated recursively with a user-defined number of recursions, $\phi \in \mathbb{N}$. As $\phi \to \infty$, the predefined normal directions begin to approximate a p-dimensional sphere, allowing the polytope to closely approximate any convex shape, at the expense of increased computational complexity. The algorithm for generating these predefined directions is outlined in Algorithm 3 and is intended to run offline. Figure 5.5 illustrates how the number of parameters p and the recursion depth ϕ influence the generation of predefined directions. As both p and ϕ increase, the density of generated directions grows, leading to a more precise approximation of the desired convex shape.

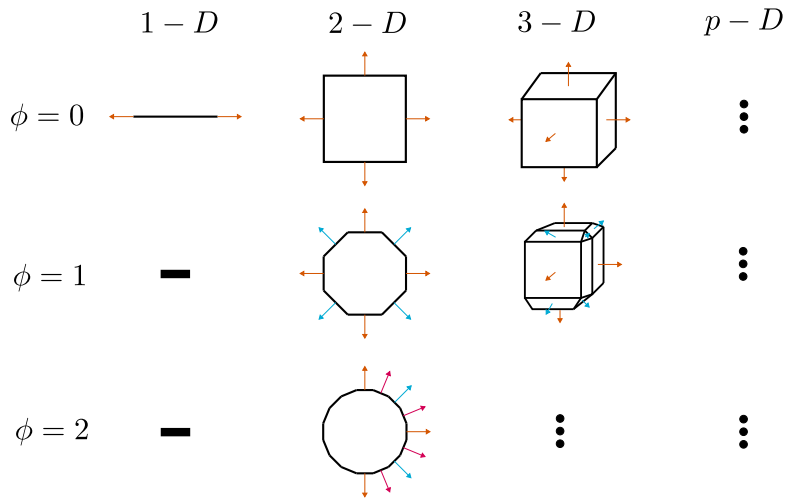

Figure 5.5: Schematic representation of the generated predefined directions $\mathcal{E}$ depending on the parameter space dimension p (problem-specific) and the user-defined iteration for accuracy in representation ϕ.

The set of vertices $\mathcal{V}_t$ of the convex polytope $\mathbf{\Theta}_t$ is computed by first removing redundant inequalities with a set of Linear Programs (LPs) solved with [178] and then by computing the solution of all possible combinations of the linear algebraic equations that describe the non-redundant inequalities. The solutions that also satisfy the inequalities are stored as the vertices of the convex polytope. Having the set predefined directions $\mathcal{E}$ from Algorithm 3 and the set of vertices $\mathcal{V}_t$ of the convex polytope $\mathbf{\Theta}_t$, we need to outer-approximate the FPS given in (5.17) at each timestep t such that $\mathbf{\Theta}_t \subseteq \overline{\mathbf{\Theta}}_t$. In contrast to [172] where an optimization problem is solved, we use linear algebra to compute the extremum vertices of the polytopic set $\mathbf{\Theta}_t$ along the directions $\mathcal{E}$. This process is described in Algorithm 4.

Having the set predefined directions $\mathcal{E}$ from Algorithm 3 and the set of vertices $\mathcal{V}$ of the convex polytope $\mathbf{\Theta}_t$, we need to outer-approximate the FPS given in (5.17) at each timestep t with an outer approximation such that $\mathbf{\Theta}_t \subseteq \overline{\mathbf{\Theta}}_t$. In contrast to [172] where an optimization problem is solved, we use simple linear algebra calculations to find the

Algorithm 3 Generate Predefined Directions (Offline)

Input: p , ϕ
Output: $\mathcal{E}$

$\mathcal{E} \leftarrow \{\pm \boldsymbol{e}_i \,:\, i = 1, \ldots, p\}$
for $i \leftarrow 1$ **to** ϕ **do**
  $\mathcal{E}^{\text{new}} \leftarrow \emptyset$
  for $j \leftarrow 1$ **to** p **do**
    for each combination of j elements in $\mathcal{E}$ **do**
      $\boldsymbol{e}^{\text{new}} \leftarrow$ sum of the combination
      if $\|\boldsymbol{e}^{\text{new}}\| \neq 0$ **then**
        $\boldsymbol{e}^{\text{new}} \leftarrow \frac{\boldsymbol{e}^{\text{new}}}{\|\boldsymbol{e}^{\text{new}}\|}$
        $\mathcal{E}^{\text{new}} \leftarrow \mathcal{E}^{\text{new}} \cup \{\boldsymbol{e}^{\text{new}}\}$
      end if
    end for
  end for
  $\mathcal{E} \leftarrow \mathcal{E}^{\text{new}}$
end for

extremum vertices of the polytopic set $\boldsymbol{\Theta}_t$ along the directions $\mathcal{E}$. This process is described in Algorithm 4.

Algorithm 4 Outer-approximate Convex Polytope

Input: $\mathcal{E}, \mathcal{V}$
Output: $\overline{\boldsymbol{\Theta}}_t$ as the pair $\langle \overline{\boldsymbol{H}}_{\boldsymbol{\theta}_t}, \overline{\boldsymbol{h}}_{\boldsymbol{\theta}_t} \rangle$

for $\boldsymbol{e}_i \in \mathcal{E}$ **do**
  $\Pi \leftarrow -\infty$
  for $\boldsymbol{v}_j \in \mathcal{V}$ **do**
    if $\boldsymbol{e}_i^\top \boldsymbol{v}_j > \Pi$ **then**
      $\Pi \leftarrow \boldsymbol{e}_i^\top \boldsymbol{v}_j$
      $\overline{\boldsymbol{v}}_i \leftarrow \boldsymbol{v}_j$
    end if
  end for
  $\overline{\boldsymbol{H}}_{\boldsymbol{\theta}_t}(i, :) \leftarrow \overline{\boldsymbol{v}}_i^\top$
  $\overline{\boldsymbol{h}}_{\boldsymbol{\theta}_t}(i) \leftarrow \boldsymbol{e}_i^\top \overline{\boldsymbol{v}}_i$
end for
$\langle \overline{\boldsymbol{H}}_{\boldsymbol{\theta}_t}, \overline{\boldsymbol{h}}_{\boldsymbol{\theta}_t} \rangle \leftarrow$ remove_redundant_constraints$(\overline{\boldsymbol{H}}_{\boldsymbol{\theta}_t}, \overline{\boldsymbol{h}}_{\boldsymbol{\theta}_t})$

5.6 Parameter Estimate

The final step is to derive an estimate $\hat{\boldsymbol{\theta}}_t$ for the unknown parameter $\boldsymbol{\theta}_t$ that belongs to the derived parameter set $\overline{\boldsymbol{\Theta}}_t$. We can exploit again here the fact that the system is linear to the parameters of interest and use the following equation:

$$\boldsymbol{G}(\boldsymbol{u}_{t-1})\boldsymbol{\theta} = \boldsymbol{y}_t - \boldsymbol{f}(\boldsymbol{y}_{t-1}) \tag{5.21}$$

which is a linear algebraic equation to the unknown parameter $\boldsymbol{\theta}_t$ and where the disturbance and the noise are included in the measurement. If we concatenate (5.21) for the last N measurements, to leverage more data, we can get a regression equation:

$$\underbrace{\begin{bmatrix} G(\boldsymbol{u}_{t-1-N}) \\ \dots \\ G(\boldsymbol{u}_{t-1}) \end{bmatrix}}_{\Phi} \boldsymbol{\theta} = \underbrace{\begin{bmatrix} \boldsymbol{y}_{t-N} - f(\boldsymbol{y}_{t-1-N}) \\ \dots \\ \boldsymbol{y}_t - f(\boldsymbol{y}_{t-1}) \end{bmatrix}}_{\xi} \tag{5.22}$$

where $\boldsymbol{\theta}$ here is the *regressand*, Φ is the *regressor* and ξ the *observation*. The solution to the unconstrained classical Least Squares Problem (LSP) has a well-known form using the expression of the Moore-Penrose pseudo-inverse. However, here we want $\hat{\boldsymbol{\theta}}_t \in \overline{\boldsymbol{\Theta}}_t$ so a closed-form solution cannot be used and instead a Quadratic Program (QP) needs to be solved online considering the linear inequality constraints introduced from $\overline{\boldsymbol{\Theta}}_t$. Furthermore, the rank of the regressor matrix Φ—which depends on input measurements—directly impacts the solvability and quality of the solution to the parameter estimation problem as the regressor is not always guaranteed to be full rank. A rank-deficient matrix implies that not all parameters in $\boldsymbol{\theta}$ are independently estimable from the given inputs, leading to either non-unique solutions or inaccurate estimates. To address these issues, we formulate the following QP with generalized Tikhonov regularization:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \boldsymbol{\theta}^\top \boldsymbol{P} \boldsymbol{\theta} + \boldsymbol{q}^\top \boldsymbol{\theta} \\ \text{s.t.:} \quad & \overline{\boldsymbol{H}}_{\boldsymbol{\theta}_t} \boldsymbol{\theta} \leq \overline{\boldsymbol{h}}_{\boldsymbol{\theta}_t}, \\ & \boldsymbol{\theta}^0 = \boldsymbol{\theta}_t^c. \end{aligned} \tag{5.23}$$

with $\boldsymbol{P} = \Phi^\top \Phi + \Lambda$, $\boldsymbol{q} = -2(\xi^\top \Phi + \tilde{\boldsymbol{\theta}}^\top \Lambda)$, Λ the regularization matrix, $\tilde{\boldsymbol{\theta}}$ the regularization value of $\boldsymbol{\theta}$, $\boldsymbol{\theta}_t^c$ the vertex centroid of polytope $\overline{\boldsymbol{\Theta}}_t$, given as $\frac{1}{N_v} \sum_{i=1}^{N_v} \overline{\boldsymbol{v}}_i$, and $\boldsymbol{\theta}^0$ the initial guess for the solution of the QP problem. Since the regressor matrix Φ depends on a window of input measurements and has a time-varying rank condition, regularization should only be significant when the matrix approaches rank deficiency and should be negligible otherwise. Therefore, we introduce an adaptive regularization parameter Λ as an exponential decay function of the rank condition of the regressor matrix:

$$\Lambda = \bar{\Lambda} e^{-\alpha \Sigma_p} \tag{5.24}$$

where $\bar{\Lambda}$ is the maximum value of the regularization parameter matrix, α is a parameter to tune the decay rate of the function, and Σ_p is the diagonal singular value matrix that results from the compact Singular Value Decomposition (SVD) of Φ. Because of the efficiency of QP solvers along with the limited problem dimension, (bounded dimensions of $\overline{\boldsymbol{H}}_{\boldsymbol{\theta}_t} \boldsymbol{\theta} \leq \overline{\boldsymbol{h}}_{\boldsymbol{\theta}_t}$ and fixed size of parameter vector p) the QP (5.23) can be solved swiftly online with [179].

5.7 Fault Decision Logic

The different components of SME outlined in previous sections are utilized to perform FD, as summarized in Algorithm 5. As new input-output measurements are obtained, the UPS and FPS are updated, alongside the outer approximation of the FPS and the parameter

estimate. If the input is bound exploring, we can obtain minimal uncertainty. Thus, under healthy conditions, the FPS typically converges to a “healthy set" of fault parameter values around the nominal value of a healthy parameter while taking into account the disturbance and noise bounds described in Assumption 1. If, at any timestep t_F, a fault occurs, the newly computed UPS will likely no longer intersect with the current FPS, depending on the fault's severity (see Figure 5.6). This implies that the most recent data provides a set of parameters that do not belong to the “healthy FPS", leading to the conclusion that a fault has been detected.

Fault Detection Logic: If $\overline{\boldsymbol{\Theta}}_{t-1} \cap \boldsymbol{\Delta}_t = \emptyset$, then a fault is guaranteed to be detected.

A fault is detected at the first timestep t_D when the following condition occurs:

$$t_D = \min\{t \mid \overline{\boldsymbol{\Theta}}_{t-1} \cap \boldsymbol{\Delta}_t = \emptyset,\ t > t_F\} \tag{5.25}$$

where t_F indicates that timestep a fault has occurred. To isolate the fault, we follow a similar approach as for the detection, but now we need to check the projection of the FPS on the different principle axes of the parameter space first. The projection of the FPS on the principle axes of the parameter space is given by:

$$\text{Proj}_{\theta_i}(\boldsymbol{\Theta}_t) = \left[\min_{\boldsymbol{v}\in\mathcal{V}}(e_{\theta_i}^\top \boldsymbol{v}), \max_{\boldsymbol{v}\in\mathcal{V}}(e_{\theta_i}^\top \boldsymbol{v})\right] \tag{5.26}$$

where $e_{\theta_i},\ i = 1, 2, \ldots, p$ denotes the unit vectors of the orthonormal basis of the parameter space, and $\boldsymbol{v} \in \mathcal{V}$ represents the vertices of the FPS. Since the FPS is a convex set by construction, this one-dimensional projection results in an interval. We can then compare these intervals before and after fault detection.

Fault Isolation Logic: If $\text{Proj}_{\theta_i}(\boldsymbol{\Theta}_t) \cap \text{Proj}_{\theta_i}(\boldsymbol{\Theta}_{t_D-1}) = \emptyset$, then θ_i is guaranteed to be faulty.

A fault is isolated at the first timestep $t_I^i,\ i = 1, 2, \ldots, p$ when the following condition occurs:

$$t_I^i = \min\{t \mid \text{Proj}_{\theta_i}(\boldsymbol{\Theta}_t) \cap \text{Proj}_{\theta_i}(\boldsymbol{\Theta}_{t_D-1}) = \emptyset,\ t > t_D\} \tag{5.27}$$

Following fault detection and isolation, the containers are reinitialized, and the parameter set begins to converge toward a “faulty" FPS, accompanied by a new estimate for the fault parameters.

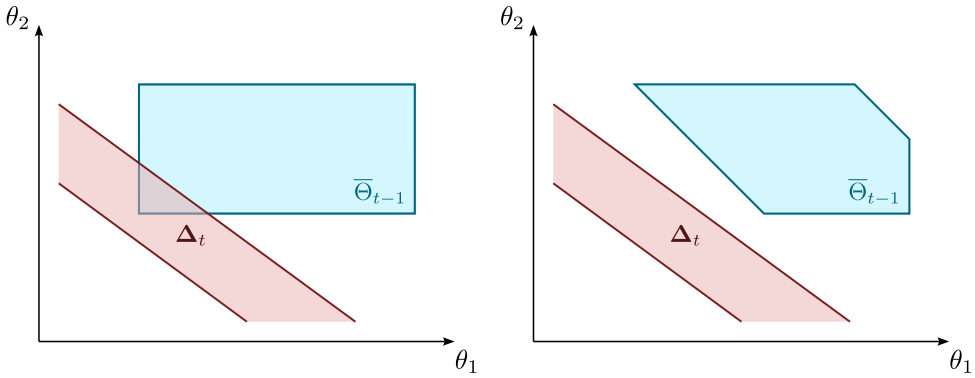

Figure 5.6: Schematic representation of fault detection using the inverse test on the FPS. In this example, the UPS arises from a measurement corrupted by a fault. In the left figure, the outer approximation of the FPS is more conservative, preventing fault detection, as $\overline{\Theta}_{t-1} \cap \Delta_t \neq \emptyset$. In the right figure, due to a tighter outer approximation of the FPS, the same measurement results in $\overline{\Theta}_{t-1} \cap \Delta_t = \emptyset$, successfully revealing the fault.

5

Algorithm 5 SME-based Fault Diagnosis

Input: $\bar{d}$, $\bar{n}$, Θ_0, $f(\cdot)$, $G(\cdot)$, ϕ
Output: $\overline{\Theta}_t$, $\hat{\theta}_t \in \overline{\Theta}_t$

Compute predefined directions, $\mathcal{E}$, from Algorithm 3
for $t = 1, 2, \ldots$ **do**
  Get input-output data $\{y_t, y_{t-1}, u_t\}$
  Compute UPS, Δ_t, from (5.19)
  Compute FPS, Θ_t, from (5.20)
  Compute projection of Θ_t, from (5.26)
  Check feasibility of FPS, Θ_t, (with LPs)
  if $\Theta_t = \emptyset$ **then**
    Fault detected
    $t_D = t$
    for $i = 1, 2, \ldots, p$ **do**
      if $\text{Proj}_{\theta_i}(\Theta_{t_D}) \cap \text{Proj}_{\theta_i}(\Theta_{t_D - 1}) = \emptyset$ **then**
        Fault isolated at θ_i
        $t_I^i = t$
      end if
    end for
    $\Theta_t = \Theta_0$
    $\Phi = [\,]$, $\xi = [\,]$
  end if
  Compute outer-approximation, $\overline{\Theta}_t$, from Algorithm 4
  Compute vertex centroid as $\theta_t^c = \frac{1}{N_v} \sum_{i=1}^{N_v} \overline{v}_i$
  Compute estimate , $\hat{\theta}_t \in \overline{\Theta}_t$, from (5.23)
end for

5.8 Results

In this case study we consider the 3-DoF model of an ASV in planar motion, equipped with sensors, actuators, and the trajectory optimizer from [166] for path-following and collision avoidance. The ASV dynamics follow the maneuvering model in [159]. Its state includes position $\boldsymbol{p} = (x, y)^\top$, orientation ψ, longitudinal velocity u, lateral velocity v, and yaw rate r, expressed in the body-fixed frame and denoted as $\boldsymbol{x} = (x, y, \psi, u, v, r)^\top \in \mathcal{Z} \subset \mathbb{R}^6$, while the control input $\boldsymbol{u} = (\tau_l, \tau_r, \tau_b, \alpha_l, \alpha_r)^\top \in \mathcal{U} \subset \mathbb{R}^5$ represents the actions of two azimuth thrusters and a bow thruster. Specifically, τ_l and τ_r are the thrusts, and α_l and α_r are the azimuth angles of the left and right thrusters, while τ_b represents the thrust of the bow thruster. Environmental disturbances from wind and waves are denoted as $\boldsymbol{\tau_d}$. The system's evolution is governed by the following continuous, nonlinear system:

$$\dot{\boldsymbol{x}} = \underbrace{\begin{bmatrix} \mathbf{0}_{3\times3} & \boldsymbol{R}(\boldsymbol{x}) \\ \mathbf{0}_{3\times3} & -\boldsymbol{M}^{-1}(\boldsymbol{C}(\boldsymbol{x}) + \boldsymbol{D}(\boldsymbol{x})) \end{bmatrix} \boldsymbol{x}}_{f(\boldsymbol{x})} + \underbrace{\begin{bmatrix} \mathbf{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix} \boldsymbol{\tau}(\boldsymbol{u})}_{g(\boldsymbol{u})} + \underbrace{\begin{bmatrix} \mathbf{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix} \boldsymbol{\tau_d}}_{\boldsymbol{d}} \tag{5.28}$$

Here, $\boldsymbol{R}(\boldsymbol{x})$ is the rotation matrix, $\boldsymbol{M}$ is the mass matrix, $\boldsymbol{C}(\boldsymbol{x})$ is the Coriolis and centripetal matrix and $\boldsymbol{D}(\boldsymbol{x})$ the damping matrix. The generalized force vector acting on the ASV is denoted by $\boldsymbol{\tau}$, and w_{lr}, l_{lr}, and l_b are length parameters that define the thruster configuration. The thrust generated by the actuators under healthy conditions is represented by $\boldsymbol{\tau}(\boldsymbol{u})$, and actuator limitations are also taken into account. For fault modeling, we introduce actuator faults represented by $\boldsymbol{\theta} = (\theta_l, \theta_r, \theta_b)^\top$, where θ_l, θ_r, and θ_b denote the loss of effectiveness (LoE) in the left, right, and bow thruster, respectively expressed in polytopic form:

$$\begin{bmatrix} \boldsymbol{I}_p & -\boldsymbol{I}_p \end{bmatrix} \boldsymbol{\theta} \leq \begin{bmatrix} \mathbf{1} & \mathbf{0} \end{bmatrix} \tag{5.29}$$

The input map can be written linearly to the parameters as:

$$\boldsymbol{g}(\boldsymbol{u}) = \underbrace{\begin{bmatrix} \mathbf{0}_{3\times3} \\ \boldsymbol{M}^{-1} \end{bmatrix} \begin{bmatrix} \tau_l \cos\alpha_l & \tau_r \cos\alpha_r & 0 \\ \tau_l \sin\alpha_l & \tau_r \sin\alpha_r & 0 \\ -w_{lr}\tau_l \cos\alpha_l & w_{lr}\tau_r \cos\alpha_r & l_b \tau_b \\ -l_{lr}\tau_l \sin\alpha_l & -l_{lr}\tau_r \sin\alpha_r & \end{bmatrix}}_{G(\boldsymbol{u})} \underbrace{\begin{pmatrix} \theta_l \\ \theta_r \\ \theta_b \end{pmatrix}}_{\boldsymbol{\theta}} \tag{5.30}$$

We assume full-state measurements with additive measurement noise. The disturbances and measurement noise follow a uniform distribution with known bounds chosen as $\bar{\boldsymbol{d}} = (0.02, 0.03, 0.003, 0.02, 0.03, 0.01)^\top$ and $\bar{\boldsymbol{n}} = (0.01, 0.01, 0.001, 0.007, 0.005, 0.012)^\top$ respectively. The dynamics in (5.28) are discretized using Runge-Kutta for numerical implementation. Our framework is implemented in ROS: the controller and FD module in C++ and the simulator in Python. The algorithm runs in an Ubuntu machine with an Intel i7 CPU@1.8GHz and 16GB of RAM.

We begin by simulating the ASV following a sinusoidal reference path seen in Figure 5.7 under healthy conditions to compare the formulation developed in this work with existing approaches that ignore noise bounds, leading to false alarms. Figure 5.8 illustrates the evolution of two different FPSs at six equidistant time instances. In blue, we represent the FPS using the proposed UPS formulation introduced here, which accounts for measurement noise. In orange, we show the FPS from the existing UPS formulation that does not consider noise. Our formulation ensures the FPS remains feasible in healthy conditions, converging

towards a "healthy area" around the healthy value $\boldsymbol{\theta} = (1\ 1\ 1)^\top$. In contrast, the previous formulation leads to several instances where the FPS becomes infeasible, preventing consistent convergence to the "healthy area". This effect is even more pronounced in Figure 5.9, where the time evolution of each fault parameter θ_i is shown, along with the corresponding set bounds as shaded areas of the same color. It is clear that the orange FPS in Figure 5.9 becomes infeasible multiple times, triggering false alarms. In contrast, our formulation is designed to prevent false alarms entirely and the set in blue converges monotonically to the healthy region. We also simulate a traffic scenario where the ASV needs to follow a

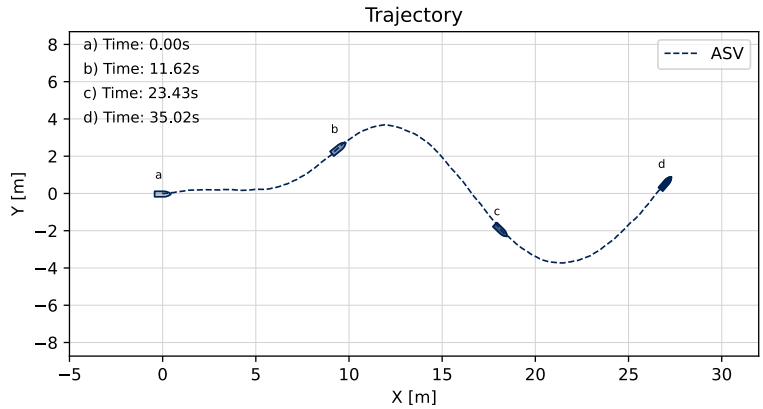

Figure 5.7: The ASV trajectory in healthy conditions while following a sinusoidal path.

straight, horizontal reference path while avoiding collisions with Obstacle Vessels (OVs) seen in Figure 5.10. The system is subjected to both disturbance and noise and at time $t_F = \Delta t \cdot t_f = 20s$, a permanent fault $\theta_r = 0.2$ is injected in the right thruster. We compare different outer approximations of the FPS and evaluate their sensitivity, as well as compare the regularized parameter estimate proposed here with the conventional one. Both FPSs are constructed from UPSs that account for measurement noise. Figure 5.11 shows the evolution of the FPS using two different outer approximations. The cyan line represents a "tighter" outer approximation from Algorithm 4 with $\phi = 1$, while the pink line illustrates a "looser" approximation with $\phi = 0$ (a simple bounding box). At time $t = 20.41s$, after the fault occurs, it is evident that the tighter approximation in cyan, being more sensitive, leads to an infeasible set, indicating a fault as it starts converging towards a "faulty area". The looser approximation in pink also converges towards the faulty region but with some delay due to its larger volume. Figure 5.12 shows the parameter estimates for this scenario. In cyan, the regularized parameter estimate is displayed, while the un-regularized estimate ($\boldsymbol{\Lambda} = \mathbf{0}$) is shown in pink. The corresponding FPS bounds are shown in matching colors. The moment of the fault is marked by a red dashed vertical line. The fault is detected at $t = 20.56s$ using the tighter approximation (blue dashed vertical line) and at $t = 21s$ using the looser approximation (purple dashed vertical line). The regularized parameter estimate (cyan continuous line) is generally more stable than the unregularized one (pink continuous line), making it a more reliable nominal estimate to evaluate the true fault value.

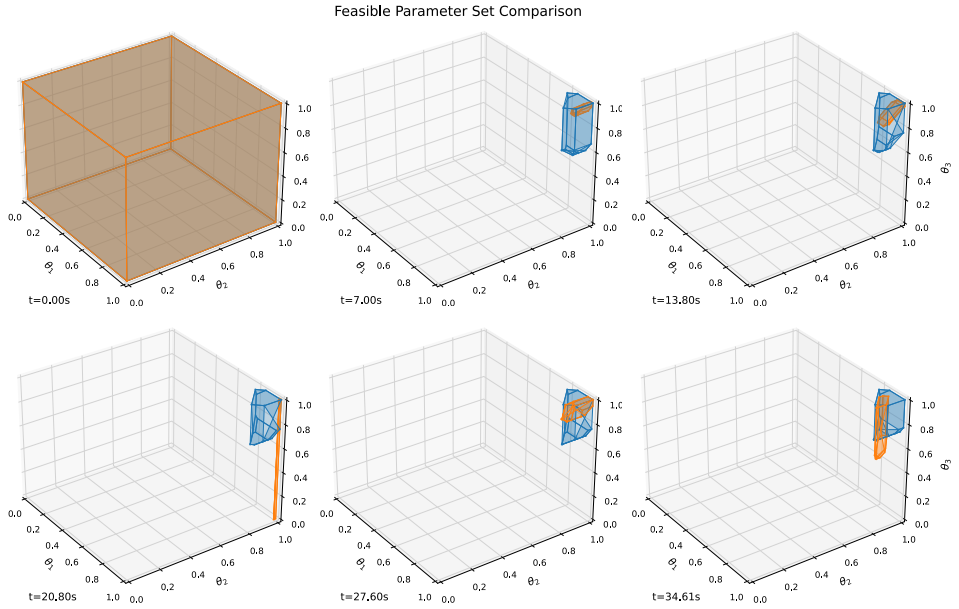

Figure 5.8: Evolution of the FPS in healthy conditions. In blue, the FPS considers measurement noise, converging towards the "healthy" region. In contrast, the orange FPS, which neglects noise, becomes infeasible multiple times and fails to converge uniformly.

The evolution of the FPS along with the ASV trajectories for both simulation experiments can be viewed in animated form in the video available at [180].

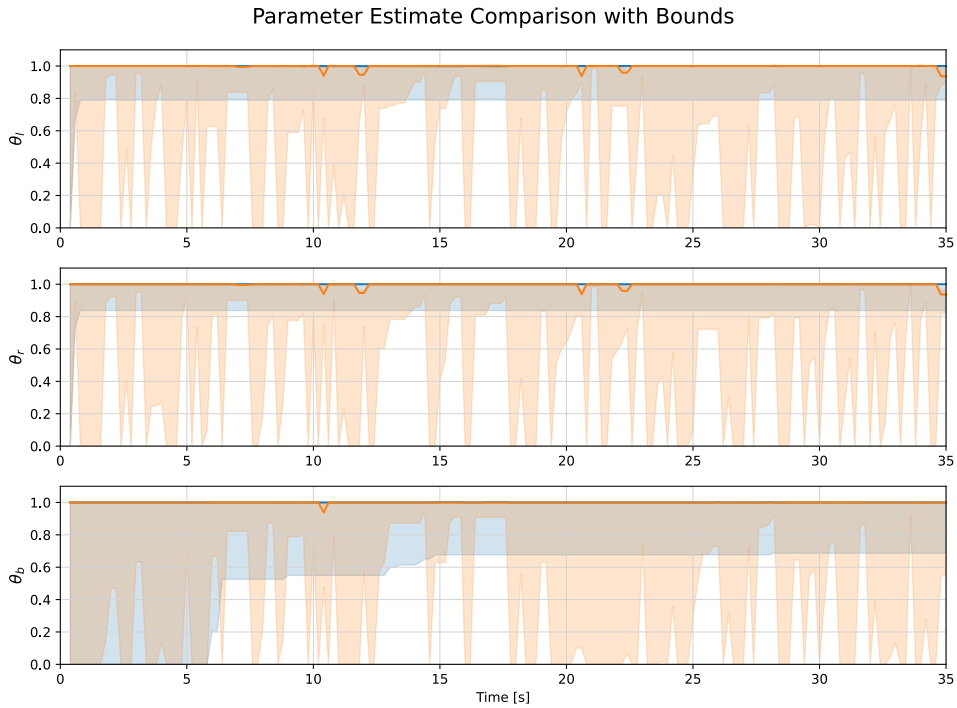

Figure 5.9: Evolution of the parameter estimate along with the corresponding bounds of the FPS projected in one dimension. The orange set, which does not account for measurement noise, becomes infeasible in several instances, triggering false alarms.

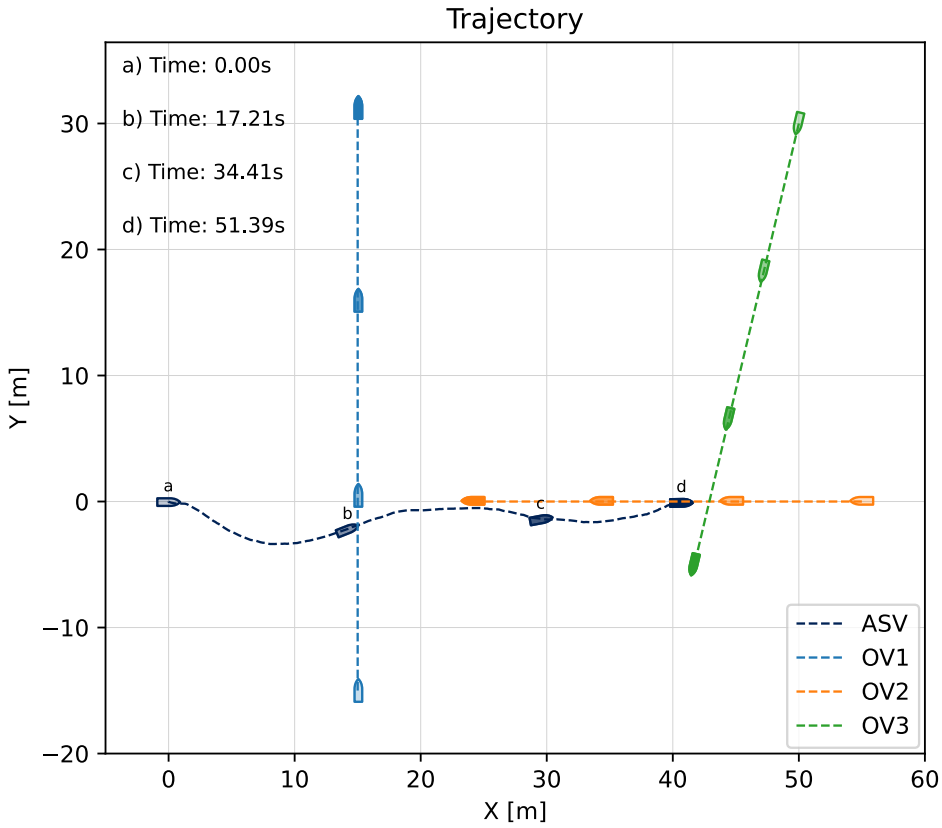

Figure 5.10: The ASV trajectory in faulty conditions while following a straight line path and avoiding collisions with the OVs.

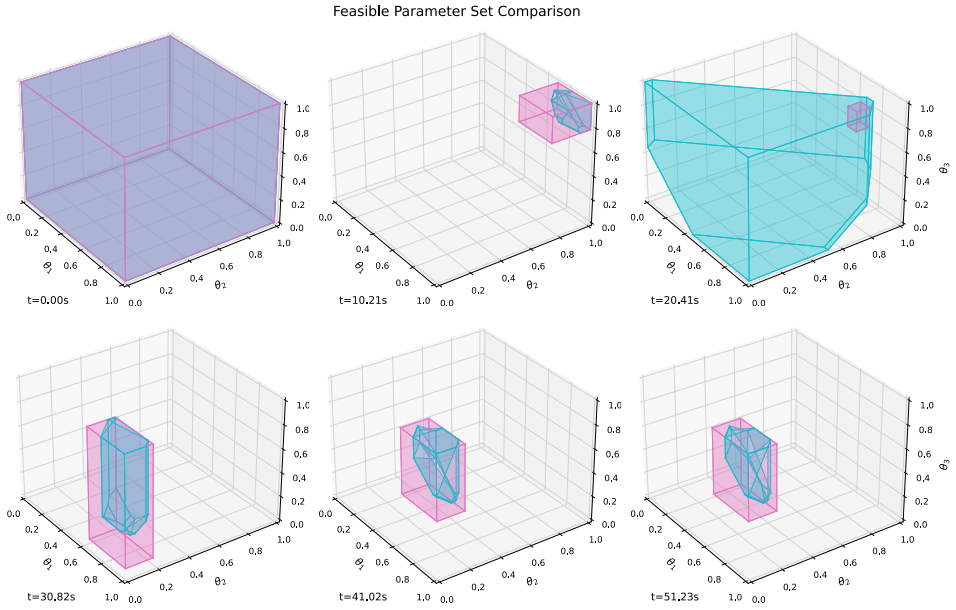

Figure 5.11: Comparison of the FPS evolution using two different outer approximations. The top-right sub-figure highlights the moment just after the fault occurs. The tighter outer approximation (cyan) detects the fault faster and begins to converge toward the faulty region, while the looser approximation (pink) converges more slowly.

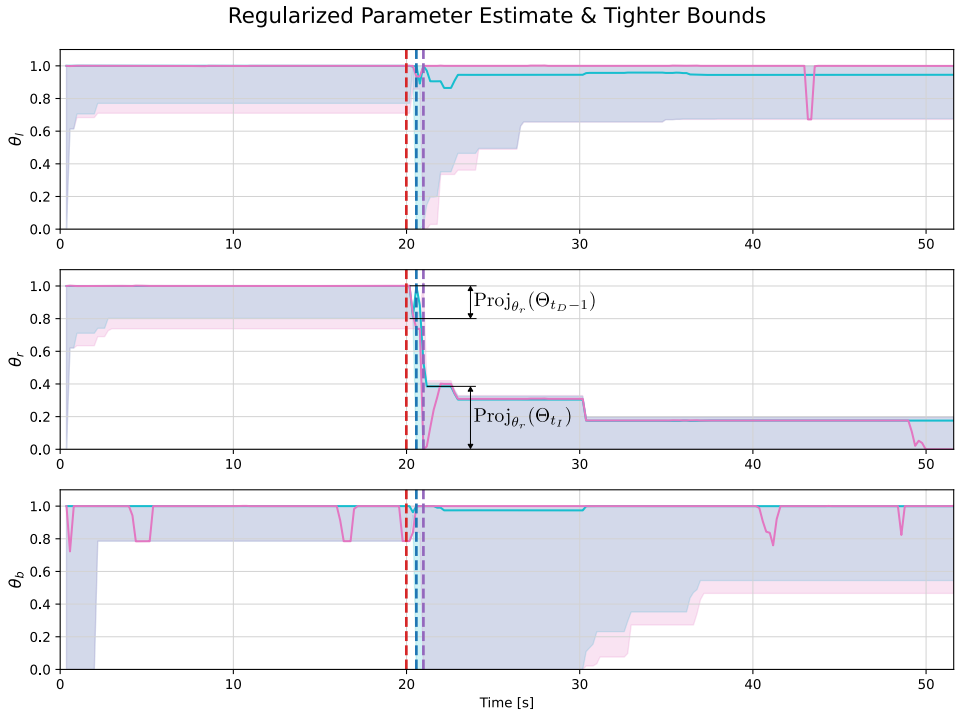

Figure 5.12: Parameter estimate with corresponding bounds (shaded areas). In the second sub-plot for θ_r, the discontinuity in FPS bounds indicates the fault in the right thruster. The fault occurs at the red dashed vertical line, with the detection time shown in purple. The regularized estimate (cyan) is more stable and closer to the true value compared to the unregularized estimate (pink).

5.9 Conclusions

In this chapter, we introduced a Fault Diagnosis method using Set Membership Estimation for uncertain nonlinear systems that are linear in the fault parameters and subject to both state and output uncertainties as a response to Research Question **Q3**: "How can fault parameters be accurately and robustly estimated under varying operational conditions, including the presence of disturbances and noise?" Our approach enhances fault diagnosis by addressing both uncertainty types, improving robustness and accuracy. It employs an inverse test for reliable fault detection and isolation, continuously refining a feasible set for fault parameter estimation. Adaptive regularization in the estimation process provides greater precision, especially when input-output data are sparse, supporting fault identifiability. Our method effectively handles both state and output uncertainties, enabling the natural detection of faults while providing accurate estimates of both a nominal value and a set of possible fault parameter values. This functionality is critical for the next chapter, where we will build upon this approach to improve the robustness of our trajectory planner, discussed in Chapter 3, significantly enhancing the system's reliability and safety in the presence of faults.

6

Fault-Tolerant Trajectory Optimization & Control

*This chapter presents a fault-tolerant trajectory optimization and control framework tailored for uncertain nonlinear systems that are linear in fault parameters and subjected to additive state uncertainties as a response to Research Question **Q4**: "How can we jointly guarantee fault-tolerant and rule-compliant trajectories for ASVs operating in mixed-traffic environments?" Building on the fault diagnosis methodologies introduced earlier, this framework combines rule-compliant collision avoidance constraints with a dual-plan strategy comprising a primary and contingency trajectory. The approach ensures real-time adaptability and safety by solving both plans simultaneously in a receding horizon manner while sharing the initial input. This ensures that a fail-safe trajectory is always available in case of fault detection, enabling robust and rule-compliant navigation even in the presence of faults and uncertainties. The method leverages incremental stabilizability concepts, precomputed scalar bounds, and adaptive terminal constraints to achieve computational efficiency without sacrificing robustness. This chapter is structured as follows: Section 6.1 introduces the topic of this chapter. Section 6.2 defines the problem formulation. Sections 6.3 to 6.8 detail the proposed fault-tolerant trajectory optimization method. We conclude with final remarks in Section 6.10.*

Parts of this chapter are under preparation for a journal manuscript co-authored by A.Tsolakis, D. Benders, J. Kohler, L. Ferranti, V. Reppa, and R.R. Negenborn

6.1 Introduction

As autonomous systems become integral to safety-critical applications, ensuring that they are fault-tolerant is paramount, particularly for systems operating in environments shared with humans. ASVs are increasingly being deployed for a wide range of complex tasks, including autonomous transportation, search-and-rescue missions, and environmental monitoring. The critical nature of these applications places stringent demands on safety and reliability, as any failure can have severe consequences. Failures in components such as actuators, sensors, or computational units can undermine operational integrity, potentially leading to system downtime, mission failure, or catastrophic accidents. Therefore, robust and fault-tolerant planning frameworks are essential to ensure safe and reliable operation in such scenarios.

In Chapter 3, we introduced an MPC-based approach to ensure rule-compliant navigation in maritime environments. The method relied on the paradigm of MPCC for the task of following a time-invariant reference path while avoiding collision with OVs according to the maritime traffic rules known as COLREGs. While the method effectively integrates traffic rules and avoids collisions under nominal conditions with multiple OVs, its ability to guarantee collision avoidance under the influence of disturbances or faults is limited and relies only on the inherent robustness of nominal MPC but without any guarantees. Real-world deployment of such systems demands a framework capable of addressing such contingencies to ensure uninterrupted and safe operation.

In Chapter 5, we developed an FD method based on SME, enabling accurate detection, isolation, and quantification of faults. By estimating the feasible parameter set and a nominal fault parameter in real time, the method provides robust insights into the nature and extent of faults via a fault decision logic that relies on inverse tests of the feasible parameter set. While the method proved effective for FD it was limited only to diagnosis without exploiting these results in an active manner for fault-tolerant control.

The results of Chapters 3 and 5 pave the way for combining fault diagnosis with trajectory optimization to develop a comprehensive fault-tolerant planning framework. Fault-tolerant algorithms in robotics aim to integrate fault diagnosis and control seamlessly. For our particular problem, RAMPC stands out among these algorithms due to its unique combination of guarantees: robustness against uncertainties, dynamic feasibility, and adaptability to faults while maintaining compliance with operational rules. With RAMPC we can leverage fault information from our FD module to adjust plans dynamically, offering an efficient, unified framework for robust and adaptive operation, particularly in environments with stringent safety requirements.

Robust MPC and Robust-Adaptive MPC methods have been widely explored for handling uncertainties in nonlinear systems. However, these approaches often struggle with computational complexity or conservatism. For instance, frameworks like [181] introduce additional state and input variables to define ellipsoidal tubes and feedback gains, offering reduced conservatism but significantly increasing the computational burden, particularly in high-dimensional systems. Conversely, simpler frameworks, though more computationally efficient, can be slightly more conservative. Similarly, the Lipschitz-bounded approach in [182], extended in [183] for state-dependent uncertainties, constructs tubes based on Lipschitz bounds, but this can lead to excessive conservatism compared to methods utilizing incremental stabilizability bounds, as demonstrated in [184]. Additionally, many

robust MPC schemes rely on constant bounds for disturbances [185, 186, 187], which limits their ability to address dynamic uncertainties directly, an area where solutions like error bounding systems [188] offer partial remedies. Despite these advances, effectively handling dynamic model mismatches remains an open challenge. In [189, 190, 191], a simple approach uses scaled geometric shapes to represent uncertainty, making the method computationally efficient and easy to implement. However, this simplicity often leads to overly conservative predictions, limiting its effectiveness over longer planning horizons. Some robust adaptive MPC methods use box-shaped representations of uncertainty, with adjustments made dynamically to account for system behavior [192, 193]. Others employ more flexible approaches, such as interval arithmetic [194], which improve upon traditional methods but can still face challenges like unbounded growth over longer planning horizons. To address this, some techniques rely on pre-computed robust sets [187, 186], ensuring stability but often requiring significant offline computation.

The main problem with the nature of faults is that the monotonicity and non-increasing properties of the FPS are no longer valid. This is because, with SME, fault detection and isolation rely on the infeasibility of the FPS when a fault occurs and its re-initialization so that it converges in another "faulty" region. Thus, a RAMPC approach similar to [175] is no longer suitable since they rely on the aforementioned properties for the FPS. Effects of this nature, that happen abruptly, are usually tackled with *contingency planning*. Contingency planning most commonly refers to a contingency, backup, or fail-safe plan that is ready to be executed in case an unexpected event occurs. In the context of motion planning, contingency planning has been utilized to address different problems so far, mainly with Contingency Model Predictive Control (CMPC). In [195, 196, 197] the main focus is uncertainties that are caused by other traffic participants while the controlled system is assumed to operate in nominal conditions. Instead, in [198, 199] the authors tackle the problem of changing operating conditions for the system. Specifically in [198] they use MPC for contingent planning in order to address a change in the friction coefficient due to icy road conditions. In [199] the authors address the problem of faults with contingency planning where they use a finite number of faulty models in an RMPC formulation. However, the key challenge is that the time that a fault occurs cannot be predicted so they must deal with a family of models all at once.

The goal of this chapter is to integrate the SME-based FD approach developed in Chapter 5 into the rule-compliant MPC framework introduced in Chapter 3, resulting in a fault-tolerant and rule-compliant trajectory optimization algorithm. To achieve this, we design a novel Contingency-, Robust-Adaptive- Model Predictive Control (CRAMPC) trajectory optimizer that can robustly plan against all disturbance and fault realizations for both a primary plan and a contingency plan. The primary plan follows directly from the formulation in [175] while leveraging the FD results from Chapter 5 to adapt dynamically to the online-estimated fault parameters. Conversely, the contingency plan is designed to account for the worst-case fault realization, ensuring that a fail-safe alternative is always available. To guarantee a smooth transition from the "healthy" to the "faulty" model without infeasibilities, the first input of the input sequences in both plans is constrained accordingly. By enforcing this constraint, as in [198], we ensure that the system can safely transition to a fault-tolerant trajectory even when an unexpected fault occurs. This integrated approach effectively addresses the challenges posed by faults, providing a robust and practical

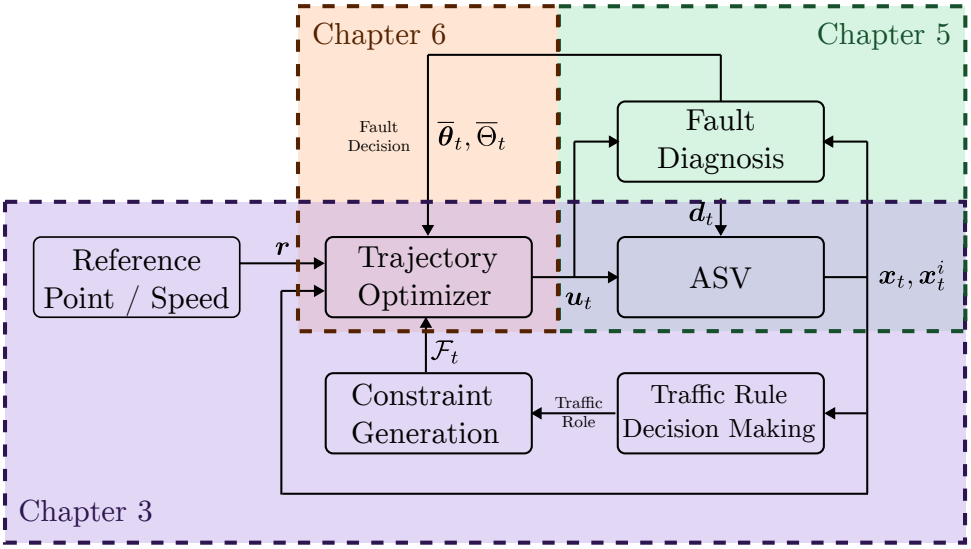

Figure 6.1: A block diagram scheme to describe the combination of the different modules developed in this Thesis. We consider the traffic rule decision-making and rule-compliant constraints module from Chapter 3 (purple) to guarantee rule-compliant trajectories for the ASV. We also consider the FD method from Chapter 5 (green) in order to estimate online the condition of the ASV (healthy or faulty, location and magnitude of faults). We then utilize these results by proposing a novel trajectory optimizer in Chapter 6 (orange) that is able to generate both rule-compliant and fault-tolerant trajectories by combining the paradigms of CMPC and RAMPC.

6

solution for ASVs operating in dynamic, uncertain, and fault-prone environments. By combining rule-compliant navigation from Chapter 3 with fault diagnosis from Chapter 5 within the proposed CRAMPC framework, the proposed method aims to ensure robust and safe navigation among other traffic participants. The framework is illustrated in a schematic overview in Figure 6.1 aiming for rule-compliant and fault-tolerant trajectory optimization and control.

6.2 Problem Formulation

We consider the discrete, nonlinear, perturbed system:

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{d}_t, \boldsymbol{\theta}) \tag{6.1}$$

where $\boldsymbol{x}_t \in \mathbb{R}^{n_x}$ is the state, $\boldsymbol{u}_t \in \mathbb{R}^{n_u}$ is the input, $\boldsymbol{d}_t \in \mathbb{D} \subset \mathbb{R}^{n_x}$ is the unknown but bounded state disturbance, and $\boldsymbol{\Theta}_t \subset \mathbb{R}^{n_\theta}$ is the time-invariant, unknown but bounded fault parameter. The elements $[\boldsymbol{\theta}]_i \in [0,1]$, $i = 1,2,\dots,n_\theta$, describe the health of the system according to the following conditions:

$$[\boldsymbol{\theta}]_i = \begin{cases} [\boldsymbol{\theta}]_i = 1, \ \forall i \in \{1,2,\dots,n_\theta\}, & \text{healthy system} \\ [\boldsymbol{\theta}]_i < 1, \ \exists i \in \{1,2,\dots,n_\theta\}, & \text{faulty system} \end{cases} \tag{6.2}$$

The bounds of the fault parameter $\boldsymbol{\theta}$ are computed online along with a fault parameter estimate $\bar{\boldsymbol{\theta}}_t \in \overline{\boldsymbol{\Theta}}_t$, updated at each timestep t based on input-state data according to the

method of Chapter 5. We further assume that the fault happens abruptly and only single faults occur. Therefore, we also consider the worst-case faulty system as follows:

$$\underset{\sim}{\boldsymbol{x}}_{t+1} = \boldsymbol{f}_{\boldsymbol{w}}(\underset{\sim}{\boldsymbol{x}}_t, \underset{\sim}{\boldsymbol{u}}_t, \boldsymbol{d}_t, \underset{\sim}{\boldsymbol{\theta}}) \tag{6.3}$$

where $\underset{\sim}{\boldsymbol{\theta}} \in \boldsymbol{\Theta}_0$ and with the set $\boldsymbol{\Theta}_0$ computed a priori as the worst-case uncertainty set based on the assumptions on the fault parameters given by:

$$\boldsymbol{\Theta}_0 = \left\{ [\boldsymbol{\theta}]_i \in [0,1] \;\middle|\; \sum_{i=1}^{n_\theta} [\boldsymbol{\theta}]_i \geq n_\theta - 1 \right\} \tag{6.4}$$

The system has constraints on the input and state that model physical and actuator limitations, expressed by a compact polytopic set $\mathcal{Z}$:

$$\mathcal{Z} := \mathcal{X} \times \mathcal{U} = \left\{ (\boldsymbol{x}, \boldsymbol{u}) \in \mathbb{R}^{n_x + n_u} \mid \boldsymbol{h}^s(\boldsymbol{x}, \boldsymbol{u}) \leq \boldsymbol{0} \right\} \tag{6.5}$$

where $\boldsymbol{h}^s(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{L}^s \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{u} \end{pmatrix} - \boldsymbol{l}^s, \boldsymbol{L}^s \in \mathbb{R}^{n_s \times (n_x + n_u)}, \boldsymbol{l}^s \in \mathbb{R}^{n_s}$.

The system has additional constraints in order to avoid collision with other traffic participants. These are expressed with respect to the position of the ASV, denoted as $\boldsymbol{p} \in \mathbb{R}^{n_p}$ with $\boldsymbol{p} = \boldsymbol{C}\boldsymbol{x}$ and $\boldsymbol{C} \in \mathbb{R}^{n_p \times n_x}$ a selection matrix for the corresponding states of $\boldsymbol{x}$. The rule-compliant collision avoidance constraints are given also in polytopic form, according to Chapter 3:

$$\mathcal{F} := \left\{ \boldsymbol{p}_{k|t} \in \mathbb{R}^{n_p} \mid \boldsymbol{h}^o_{k|t}(\boldsymbol{p}_{k|t}) \leq 0 \right\} \tag{6.6}$$

where $\boldsymbol{h}^o_{k|t}(\boldsymbol{p}_{k|t}) = \boldsymbol{L}^o_{k|t}\boldsymbol{p}_{k|t} - \boldsymbol{l}^o_{k|t}, \boldsymbol{L}^o_{k|t} \in \mathbb{R}^{n_o \times n_p}, \boldsymbol{l}^o_{k|t} \in \mathbb{R}^{n_o}$ with subscript k denoting the time index in the MPC problem described shortly.

We formulate the following *contingency optimization problem* similar to [198] that consists of two plans: a *primary plan* that uses the *primary model* in (6.1) and a *contingency plan* that uses the worst-case-fault, *contingency model* of (6.3). Each plan is formulated according to the RAMPC approach in [175] to be robust to all possible fault and disturbance realizations. We assume that at each time step t, given the measured state $\boldsymbol{x}_t$, and an online estimated fault parameter and feasible parameter set $\overline{\boldsymbol{\Theta}}_t$, the optimization problem is given

by:

$$\min_{\substack{\bar{\boldsymbol{u}}_{\cdot|t},\bar{w}_{\cdot|t}\\ \underset{\sim}{\boldsymbol{u}}_{\cdot|t},\underset{\sim}{w}_{\cdot|t}}} \sum_{k=0}^{N-1} \ell\left(\bar{\boldsymbol{x}}_{k|t},\bar{\boldsymbol{u}}_{k|t},\boldsymbol{r}_{k|t}\right) + V_f\left(\bar{\boldsymbol{x}}_{N|t}\right) + \tilde{\ell}\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{k|t},\underset{\sim}{\bar{\boldsymbol{u}}}_{k|t},\underset{\sim}{\boldsymbol{r}}_{k|t}\right) + \tilde{V}_f\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{N|t}\right) \tag{6.7a}$$

s.t.:

$$\bar{\boldsymbol{x}}_{0|t} = \boldsymbol{x}_t, \quad \delta_{0|t} = 0 \qquad \underset{\sim}{\bar{\boldsymbol{x}}}_{0|t} = \boldsymbol{x}_t, \quad \underset{\sim}{\delta}_{0|t} = 0 \tag{6.7b}$$

$$\bar{\boldsymbol{x}}_{k+1|t} = \boldsymbol{f}_{\bar{\boldsymbol{\theta}}_t}(\boldsymbol{x}_t,\boldsymbol{u}_t,\bar{\boldsymbol{\theta}}_t) \qquad \underset{\sim}{\bar{\boldsymbol{x}}}_{k+1|t} = \boldsymbol{f}_{\underset{\sim}{\boldsymbol{\theta}}}(\underset{\sim}{\bar{\boldsymbol{x}}}_{t},\underset{\sim}{\bar{\boldsymbol{u}}}_{t},\underset{\sim}{\boldsymbol{\theta}}) \tag{6.7c}$$

$$\delta_{k+1|t} = \rho_{\bar{\boldsymbol{\theta}}_t}\delta_{k|t} + w_{k|t} \qquad \underset{\sim}{\delta}_{k+1|t} = \rho_{\underset{\sim}{\boldsymbol{\theta}}}\underset{\sim}{\delta}_{k|t} + \underset{\sim}{w}_{k|t} \tag{6.7d}$$

$$w_{k|t} \geq w_{\delta,\overline{\boldsymbol{\Theta}}_t,\mathbb{D}}\left(\bar{\boldsymbol{x}}_{k|t},\bar{\boldsymbol{u}}_{k|t},\delta_{k|t}\right) \qquad \underset{\sim}{w}_{k|t} \geq w_{\underset{\sim}{\delta},\underset{\sim}{\Theta},\mathbb{D}}\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{k|t},\underset{\sim}{\bar{\boldsymbol{u}}}_{k|t},\underset{\sim}{\delta}_{k|t}\right) \tag{6.7e}$$

$$[\boldsymbol{h}^{\boldsymbol{s}}]_i\left(\bar{\boldsymbol{x}}_{k|t},\bar{\boldsymbol{u}}_{k|t}\right) + \epsilon_i^{\boldsymbol{s}}\delta_{k|t} \leq 0 \qquad [\boldsymbol{h}^{\boldsymbol{s}}]_i\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{k|t},\underset{\sim}{\bar{\boldsymbol{u}}}_{k|t}\right) + \epsilon_i^{\boldsymbol{s}}\underset{\sim}{\delta}_{k|t} \leq 0 \tag{6.7f}$$

$$[\boldsymbol{h}^{\boldsymbol{o}}]_j\left(\bar{\boldsymbol{x}}_{k|t},\bar{\boldsymbol{u}}_{k|t}\right) + \epsilon^{\boldsymbol{o}}\delta_{k|t} \leq 0 \qquad [\boldsymbol{h}^{\boldsymbol{o}}]_j\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{k|t},\underset{\sim}{\bar{\boldsymbol{u}}}_{k|t}\right) + \epsilon^{\boldsymbol{o}}\underset{\sim}{\delta}_{k|t} \leq 0 \tag{6.7g}$$

$$\delta_{k|t} \leq \bar{\delta}, \quad w_{k|t} \leq \bar{w}_{\overline{\boldsymbol{\Theta}}_t} \qquad \underset{\sim}{\delta}_{k|t} \leq \underset{\sim}{\bar{\delta}}, \quad \underset{\sim}{w}_{k|t} \leq \underset{\sim}{\bar{w}}_{\underset{\sim}{\Theta}} \tag{6.7h}$$

$$\left(\bar{x}_{N|t},\delta_{N|t}\right) \in \mathcal{X}_{f,\bar{\theta}_t,\overline{\boldsymbol{\Theta}}_t} \qquad \left(\underset{\sim}{\bar{x}}_{N|t},\underset{\sim}{\delta}_{N|t}\right) \in \mathcal{X}_{f,\underset{\sim}{\theta},\underset{\sim}{\Theta}} \tag{6.7i}$$

$$\bar{\boldsymbol{u}}_{0|t} = \underset{\sim}{\bar{\boldsymbol{u}}}_{0|t} \tag{6.7j}$$

$$k = 0,\ldots,N-1, \quad i = 1,\ldots,n_{\boldsymbol{s}}, \quad j = 1,\ldots,n_{\boldsymbol{o}} \tag{6.7k}$$

The objective function (6.7a) depends on the primary nominal state $\bar{\boldsymbol{x}}_{k|t}$, input $\bar{\boldsymbol{u}}_{k|t}$, and reference $\boldsymbol{r}_{k|t}$ and the contingency nominal state $\underset{\sim}{\bar{\boldsymbol{x}}}_{k|t}$, input $\underset{\sim}{\bar{\boldsymbol{u}}}_{k|t}$, and reference $\underset{\sim}{\boldsymbol{r}}_{k|t}$ and consists of the reference-point following stage-costs $\ell\left(\bar{\boldsymbol{x}}_{k|t},\bar{\boldsymbol{u}}_{k|t}\right)$ and $\tilde{\ell}\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{k|t},\underset{\sim}{\bar{\boldsymbol{u}}}_{k|t}\right)$ and the terminal costs $V_f\left(\bar{\boldsymbol{x}}_{N|t}\right)$ and $\tilde{V}_f\left(\underset{\sim}{\bar{\boldsymbol{x}}}_{N|t}\right)$. Note that, while safety can be ensured independently of the cost, the objective function is typically designed to prioritize the primary plan, as faults are exceptions rather than the norm. In most cases, the cost associated with the contingency trajectory, $\tilde{\ell}(\cdot,\cdot)$, is set to zero or kept very small, ensuring that performance is primarily evaluated based on the primary plan. The problem is initialized with (6.7b) at the currently measured state $\boldsymbol{x}_t$ for both the primary and the contingency plan, with the nominal state evolution under nominal dynamics (6.7c), and the tube evolution (6.7d). Equation (6.7e) constitutes the worst-case, mixed-uncertainty bounds for the mixed uncertainties $w_{\cdot|t}$ and $\underset{\sim}{w}_{\cdot|t}$ respectively, which are introduced as additional decision variables for numerical reasons. The nominal dynamics, tube evolution, and mixed-uncertainty bound for the primary plan are adaptive and depend on the estimate of the fault parameter $\bar{\boldsymbol{\theta}}_t \in \overline{\boldsymbol{\Theta}}_t$ which is updated online, while for the contingency plan they are static and depend on the worst-case fault parameter $\underset{\sim}{\boldsymbol{\theta}} \in \boldsymbol{\Theta}_0$. The tightened constraints for system limitations and collision avoidance are described in (6.7f) and (6.7g) respectively where the subscripts i and j denote the corresponding rows of the matrix equations described in (6.6), (6.5). To limit conservativeness, upper bounds for the tube evaluation and the mixed uncertainty are also introduced in (6.7h). Terminal constraints that depend on the incremental Lyapunov function (as sub-level sets) are introduced in (6.7i). Lastly, in (6.7j) the two plans are constrained in their first input of the input sequence and allowed to diverge. The optimization problem (6.7) is solved online at each time step t and the input $\boldsymbol{u}_t = \bar{\boldsymbol{u}}^*_{0|t}$ is applied to the system in a closed-loop fashion as in classical MPC. Each plan has an additional decision variable $w_{\cdot|t}$, an additional state $\delta_{\cdot|t}$ and additional constraints (6.7e), (6.7h) with respect to a nominal MPC problem. A schematic representation of the plans resulting from the above MPC

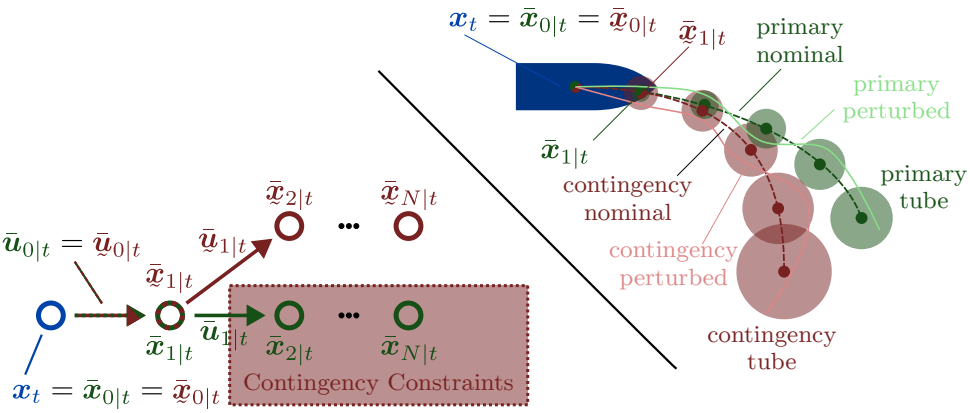

Figure 6.2: Top-right: The primary (green) and contingency (red) plans for the ASV. Both plans follow the RAMPC formulation of [175] to be robust against all possible realizations of disturbances and parametric uncertainties due to faults. Bottom-left: The primary plan (green) focuses on performance. The contingency plan (red) must obey the contingency constraints (red-shaded box). The first input, $\bar{u}_{0|t}$, is shared between the plans, optimized to meet both nominal and contingency objectives (Figure adapted from [198]).

formulation is illustrated in Figure 6.2. The primary and contingency plans are depicted, to show that they share the first input of the input sequence while they can diverge later, due to the possibly different objectives and constraints, resulting in two distinct plans. The inclusion of a contingency plan is motivated by the need to ensure recursive feasibility, providing a provably safe backup plan upon fault detection. This guarantees safety even under the worst-case realization of a fault. Ultimately, the result is a robust and provably safe plan, capable of handling the worst-case scenarios for both disturbances and faults. Note that while the contingency plan exists to make sure that there is always a fail-safe trajectory to avoid a collision even in the case of a fault, the approach is still an active FTC approach since the primary plan is adaptive and depends on the updated fault parameters.

The tube-based approach offers several advantages that make it particularly suitable for applications requiring robust performance in dynamic and uncertain environments. Unlike more conservative methods, it avoids excessive restrictions on the system's operation, while its reliance on scalar bounds ensures efficient implementation. On the other hand, the contingency approach offers the advantage of foreseeing unexpected events that may deteriorate the system's performance and lead to poor predictions of the primary plan. Therefore, the combination of these features make the proposed framework an ideal choice for scenarios where both disturbances and faults must be handled effectively, ensuring safe and reliable system performance. As such, the approach adopted in this chapter aligns well with the requirements of fault-tolerant and robust trajectory optimization and control.

6.3 Incremental Stabilizability and Offline Computations

The classical approach in tube RMPC for linear systems typically relies on the decomposition of the nominal state and the error dynamics which allows the design of an auxiliary, "tube"

feedback control law that keeps the perturbed state $\boldsymbol{x}$ close to the nominal $\bar{\boldsymbol{x}}$. The size of this tube is usually characterized by a Robust Positive Invariant (RPI) set which then leads to a predicted tube centered around the nominal trajectory where the perturbed state is guaranteed to stay inside. In general, this error decomposition is not possible for nonlinear systems since the error dynamics usually depend on the nominal trajectory and thus a different approach can be followed based on the property of incremental stabilizability.

A system is said to be incrementally stabilizable if it is possible to design a control law that ensures the distance between any two trajectories of the system decreases over time, irrespective of their initial conditions, within a certain region of interest. Incremental stabilizability then assumes that there exists an incremental Lyapunov function $V_\delta(\boldsymbol{x}, \bar{\boldsymbol{x}})$ as defined in [200]. This can be computed offline based on the solution of a semi-definite program (SDP) following the formulation in [201, 202]:

$$\min_{\boldsymbol{X},\boldsymbol{Y},\epsilon^{\boldsymbol{s}}_j,\epsilon^{\boldsymbol{o}}} \sum_{j=1}^{n^{\boldsymbol{s}}} c^{\boldsymbol{s}2}_j \epsilon^{\boldsymbol{s}}_j + c^{\boldsymbol{o}2}\epsilon^{\boldsymbol{o}} \tag{6.8a}$$

$$\text{s.t.} \quad \boldsymbol{A}(\bar{\boldsymbol{z}})\boldsymbol{X} + \boldsymbol{B}(\bar{\boldsymbol{z}})\boldsymbol{Y} + [\boldsymbol{A}(\bar{\boldsymbol{z}})\boldsymbol{X} + \boldsymbol{B}(\bar{\boldsymbol{z}})\boldsymbol{Y}]^\top + 2\rho_{\bar{\theta}_0}\boldsymbol{X} \preceq \boldsymbol{0}, \tag{6.8b}$$

$$\begin{bmatrix} \boldsymbol{A}(\bar{\boldsymbol{z}})\boldsymbol{X} + \boldsymbol{B}(\bar{\boldsymbol{z}})\boldsymbol{Y} + [\boldsymbol{A}(\bar{\boldsymbol{z}})\boldsymbol{X} + \boldsymbol{B}(\bar{\boldsymbol{z}})\boldsymbol{Y}]^\top + \lambda\boldsymbol{X} & \boldsymbol{d} \\ \boldsymbol{d}^\top & \lambda \end{bmatrix} \preceq \boldsymbol{0}, \tag{6.8c}$$

$$\begin{bmatrix} \epsilon^{\boldsymbol{s}}_j & \boldsymbol{L}^{\boldsymbol{s}}{}_{[j]} \begin{bmatrix} \boldsymbol{Y} \\ \boldsymbol{X} \end{bmatrix} \\ \left(\boldsymbol{L}^{\boldsymbol{s}}{}_{[j]} \begin{bmatrix} \boldsymbol{Y} \\ \boldsymbol{X} \end{bmatrix}\right)^\top & \end{bmatrix} \preceq \boldsymbol{0}, \quad j = 1, \dots, n_{\boldsymbol{s}}, \tag{6.8d}$$

$$\begin{bmatrix} \epsilon^{\boldsymbol{o}} & \boldsymbol{C}\boldsymbol{X} \\ (\boldsymbol{C}\boldsymbol{X})^\top & \boldsymbol{X} \end{bmatrix} \preceq \boldsymbol{0}, \quad j = 1, \dots, n_{\boldsymbol{o}}, \tag{6.8e}$$

$$\boldsymbol{X} \succeq \boldsymbol{0} \tag{6.8f}$$

$$\bar{\boldsymbol{z}} = \begin{bmatrix} \bar{\boldsymbol{x}} \\ \bar{\boldsymbol{u}} \end{bmatrix} \in \mathcal{Z}, \quad \underset{\sim}{\boldsymbol{\theta}} \in \underset{\sim}{\boldsymbol{\Theta}}. \tag{6.8g}$$

using the linearized system dynamics:

$$\boldsymbol{A}(\bar{\boldsymbol{z}}) = \left.\frac{\partial \boldsymbol{f}_{\underset{\sim}{\boldsymbol{\theta}}}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})}{\partial \bar{\boldsymbol{x}}}\right|_{(\bar{\boldsymbol{z}}, \underset{\sim}{\boldsymbol{\theta}})}, \quad \boldsymbol{B}(\bar{\boldsymbol{z}}) = \left.\frac{\partial \boldsymbol{f}_{\underset{\sim}{\boldsymbol{\theta}}}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})}{\partial \bar{\boldsymbol{u}}}\right|_{(\bar{\boldsymbol{z}}, \underset{\sim}{\boldsymbol{\theta}})}. \tag{6.9}$$

where the contraction rate $\rho_{\bar{\theta}_0} \in (0, 1)$ and parameter $\lambda \geq 0$ can be found using bi-section and the tuning parameters $c^{\boldsymbol{s}}_j, c^{\boldsymbol{o}}$ are normalized with respect to the system constraint intervals to ensure equal tightening of each constraint by using tightening constants $\epsilon^{\boldsymbol{s}}_j, \epsilon^{\boldsymbol{o}}$ for the system and obstacle constraints described in Section 6.7. The SDP (6.8) is a convex optimization problem with scalar and matrix-valued decision variables, which tries to maximize the "size" of the RPI set expressed in the objective function (6.8a), while satisfying conditions on: i) the exponential contraction of the incremental Lyapunov function (6.8b), ii) invariance (6.8c), iii) robust system constraint satisfaction (6.8d), iv) robust collision avoidance constraint satisfaction (6.8e), and v) positive definiteness of the decision variable (6.8f), expressed as Linear Matrix Inequalities (LMIs), over the gridding

(6.8g). The required elements to solve (6.8) are the linearized system dynamics (6.9), and the system constraints (6.5). Because the SDP is infinite-dimensional, it is practically solved by gridding the feasible state- and input-space $\mathcal{Z}$ and parameter-space $\overline{\boldsymbol{\Theta}}$. For variables that appear affinely, in the elements of the matrices in (6.9), it is enough to take the extremum vertices and gridding is not necessary. The solution of the SDP yields the terminal cost matrix $\boldsymbol{P}$, and the feedback law gain $\boldsymbol{K}$ computed as:

$$\boldsymbol{P} = \boldsymbol{X}^{-1}, \quad \boldsymbol{K} = \boldsymbol{Y}\boldsymbol{P} \tag{6.10}$$

The incremental Lyapunov function is then defined as a quadratic function:

$$V_\delta = \|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_{\boldsymbol{P}}^2 \tag{6.11}$$

and it will be used in the subsequent sections to derive a bound for the mixed uncertainty and the description of tube propagation in Section 6.6, and to define the terminal ingredients in Section 6.8. It also yields the tightening constants $\epsilon_j^{\boldsymbol{s}}, \epsilon^{\boldsymbol{o}}$ which will be used later in Section 6.7 for constraint tightening.

6.4 Primary and Contingency Stage Cost

The primary stage cost $\ell\left(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t}, \boldsymbol{r}_{k|t}\right)$ in (6.7a) encodes different objectives for the trajectory optimization problem expressed as a desired reference $\boldsymbol{r}_{k|t} = \left(\boldsymbol{x}^{\boldsymbol{r}\top}\ \boldsymbol{u}^{\boldsymbol{r}\top}\right)^\top$. The stage cost is then expressed in a quadratic form as:

$$\ell\left(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t}, \boldsymbol{r}_{k|t}\right) = \|\bar{\boldsymbol{x}} - \boldsymbol{x}^r\|_{\boldsymbol{Q}}^2 + \|\bar{\boldsymbol{u}} - \boldsymbol{u}^r\|_{\boldsymbol{R}}^2 \tag{6.12}$$

where $\boldsymbol{x}^r$ encodes different state-dependent objectives including following a reference point and a desired velocity, while $\boldsymbol{u}^r$ usually encodes a minimization of excessive control inputs. The matrices $\boldsymbol{Q} \succeq \boldsymbol{0}$, $\boldsymbol{R} \succ \boldsymbol{0}$, are usually diagonal and are tuned up to the desired overall performance of the trajectory optimizer (6.7).

The contingency stage cost $\tilde{\ell}\left(\tilde{\boldsymbol{x}}_{k|t}, \tilde{\boldsymbol{u}}_{k|t}, \tilde{\boldsymbol{r}}_{k|t}\right)$ in (6.7a) can be used to encode similar objectives to the ones in the primary plan expressed as a desired reference $\tilde{\boldsymbol{r}}_{k|t} = \left(\tilde{\boldsymbol{x}}^{\boldsymbol{r}\top}\ \tilde{\boldsymbol{u}}^{\boldsymbol{r}\top}\right)^\top$. The stage cost is then expressed in a quadratic form as:

$$\tilde{\ell}\left(\tilde{\boldsymbol{x}}_{k|t}, \tilde{\boldsymbol{u}}_{k|t}, \tilde{\boldsymbol{r}}_{k|t}\right) = \|\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{x}}^r\|_{\tilde{\boldsymbol{Q}}}^2 + \|\tilde{\boldsymbol{u}} - \tilde{\boldsymbol{u}}^r\|_{\tilde{\boldsymbol{R}}}^2 \tag{6.13}$$

where $\tilde{\boldsymbol{x}}^r$ also encodes state-dependent objectives including following a reference point and a desired velocity, while $\tilde{\boldsymbol{u}}^r$ usually encodes a minimization of excessive control inputs. The matrices $\tilde{\boldsymbol{Q}} \succeq \boldsymbol{0}$, $\tilde{\boldsymbol{R}} \succ \boldsymbol{0}$, are usually diagonal and are tuned up to the desired overall performance of the trajectory optimizer (6.7). Note that since faults are considered as exceptions, it is undesirable for the worst-case behavior in $\tilde{\boldsymbol{x}}$ to dominate overall performance when safety is not a concern. Typically, the focus is on optimizing the cost of the primary plan while assigning minimal weight to the cost of the contingent trajectory.

6.5 Primary and Contingency Model

In this section, we are revisiting the results from Chapter 5 and reformulating them suitably for RAMPC. We begin by considering a special class of nonlinear systems (6.1) where the

fault parameters $\boldsymbol{\theta}$ enter affinely and the disturbances are additive. The *primary-perturbed* system is given by:

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{d}_t, \boldsymbol{\theta}) = \boldsymbol{f}(\boldsymbol{x}_t) + \boldsymbol{G}(\boldsymbol{u}_t)\boldsymbol{\theta} + \boldsymbol{d}_t \tag{6.14}$$

In the following we denote by $\boldsymbol{f}_{\bar{\boldsymbol{\theta}}_t}$ the *primary-nominal* system given by:

$$\boldsymbol{x}_{t+1} = \boldsymbol{f}_{\bar{\boldsymbol{\theta}}_t}(\boldsymbol{x}_t, \boldsymbol{u}_t, \bar{\boldsymbol{\theta}}_t) = \boldsymbol{f}(\boldsymbol{x}_t) + \boldsymbol{G}(\boldsymbol{u}_t)\bar{\boldsymbol{\theta}}_t \tag{6.15}$$

with online determined parameters $\bar{\boldsymbol{\theta}}_t \in \overline{\boldsymbol{\Theta}}_t \subset \mathbb{R}^p$ where $\overline{\boldsymbol{\Theta}}_t$ denotes the Feasible Parameter Set (FPS) that is computed online according to input-state data in a SME fashion as described in Chapter 5. We first assume that the disturbances $\boldsymbol{d}_t \in \mathbb{D}$ can be written in polytopic form as:

$$|\boldsymbol{d}_t| \leq \bar{\boldsymbol{d}} \Leftrightarrow \boldsymbol{d}_t \in \mathbb{D} = \{\boldsymbol{d}_t \in \mathbb{R}^{n_x} | \boldsymbol{H}\boldsymbol{d}_t \leq \boldsymbol{h}_{\boldsymbol{d}}\} \tag{6.16}$$

where $\boldsymbol{H} = \begin{bmatrix} \boldsymbol{I}_{n_x} & -\boldsymbol{I}_{n_x} \end{bmatrix}^\top \in \mathbb{R}^{2n_x \times n_x}$ with $\boldsymbol{I}_{n_x} \in \mathbb{R}^{n_x \times n_x}$ the identity matrix of dimension $n_{\boldsymbol{x}}$, and $\boldsymbol{h}_{\boldsymbol{d}} = \begin{bmatrix} \bar{\boldsymbol{d}} & \bar{\boldsymbol{d}} \end{bmatrix}^\top \in \mathbb{R}^{2n_x}$. Combining the linearly parameterized system (6.14) with the polytopic description for the disturbance bounds in (6.16) we can compute the Unfalsified Parameter Set (UPS) at each time step:

$$\Delta_t = \{\boldsymbol{\theta} \in \mathbb{R}^p \mid -\boldsymbol{H}\boldsymbol{G}(\boldsymbol{u}_{t-1})\boldsymbol{\theta} \leq \boldsymbol{h}_{\boldsymbol{d}} + \boldsymbol{H}\boldsymbol{f}(\boldsymbol{x}_{t-1}) - \boldsymbol{H}\boldsymbol{x}_t\} \tag{6.17}$$

The FPS is then given recursively by a constant update:

$$\overline{\boldsymbol{\Theta}}_t = \overline{\boldsymbol{\Theta}}_{t-1} \cap \Delta_t \tag{6.18}$$

starting from an initially known FPS denoted as $\overline{\boldsymbol{\Theta}}_0$. In order to bound computational complexity we follow [175] where the FPS is outer-approximated by a hypercube:

$$\overline{\boldsymbol{\Theta}}_t = \bar{\boldsymbol{\theta}}_t \oplus \eta_t \mathbb{B}_\infty \tag{6.19}$$

with $\left[\bar{\boldsymbol{\theta}}_t\right]_i = (\bar{\theta}_{i,t}^{min} + \bar{\theta}_{i,t}^{max})/2$, the center of the hypercube and $\eta_t = 0.5 \max_i \left(\boldsymbol{\theta}_{i,t}^{max} - \boldsymbol{\theta}_{i,t}^{min}\right)$ its side and with $\mathbb{B}_\infty := \{\boldsymbol{\theta} \,|\, \|\boldsymbol{\theta}\|_\infty \leq 1\}$ denoting the unit hypercube.

For the contingency plan, we consider the *contingency-perturbed* system, which is subjected to disturbances and the worst-case parametric uncertainty $\underset{\sim}{\boldsymbol{\theta}} \in \underset{\sim}{\boldsymbol{\Theta}}$ with the set $\underset{\sim}{\boldsymbol{\Theta}}$ computed a priori as the worst-case uncertainty set given from (6.4):

$$\underset{\sim}{\boldsymbol{x}}_{t+1} = \boldsymbol{f}_{\underset{\sim}{\boldsymbol{w}}}(\underset{\sim}{\boldsymbol{x}}_t, \underset{\sim}{\boldsymbol{u}}_t, \boldsymbol{d}_t, \underset{\sim}{\boldsymbol{\theta}}) = \boldsymbol{f}(\underset{\sim}{\boldsymbol{x}}_t) + \boldsymbol{G}(\underset{\sim}{\boldsymbol{u}}_t)\underset{\sim}{\boldsymbol{\theta}} + \boldsymbol{d}_t \tag{6.20}$$

We also consider the *contingency-nominal* system that is used for the contingency plan in 6.7 given by:

$$\bar{\underset{\sim}{\boldsymbol{x}}}_{t+1} = \boldsymbol{f}_{\underset{\sim}{\boldsymbol{\theta}}}(\bar{\underset{\sim}{\boldsymbol{x}}}_t, \bar{\underset{\sim}{\boldsymbol{u}}}_t, \underset{\sim}{\boldsymbol{\theta}}) = \boldsymbol{f}(\underset{\sim}{\boldsymbol{x}}_t) + \boldsymbol{G}(\underset{\sim}{\boldsymbol{u}}_t)\underset{\sim}{\boldsymbol{\theta}} \tag{6.21}$$

The main difference between the primary and the contingency models is that the primary model is adaptive and updated at each time step t to give a less conservative estimation of the model. It is used in the primary plan to opt for performance. In contrast, the contingency model is static and considers only the worst-case possible fault. It is used in the contingent plan where the focus is on safety and the need for a fail-safe trajectory at all times.

6.6 Uncertainty Description and Tube Propagation

There are two main sources of uncertainty in the perturbed systems (6.14) and (6.20), namely the additive state disturbance $\boldsymbol{d}_t \in \mathbb{D}$ and the parametric uncertainty that is bounded by the polytopic sets $\overline{\boldsymbol{\Theta}}_t$ for the primary and $\tilde{\boldsymbol{\Theta}}$ for the contingency model respectively.

For the primary system, we denote the total effect of the two uncertainties with $w_{k|t}$, a scalar decision variable in the optimization problem (6.7). We compute the worst-case realization of this mixed uncertainty to use as a lower bound in (6.7e):

$$w_{\delta,\overline{\Theta}_t,\mathbb{D}}(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t}, \delta_{k|t}) = w_{\overline{\Theta}_t,\mathbb{D}}(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t}) + L_{\overline{\Theta}_t}\delta_{k|t} \tag{6.22}$$

with $w_{\overline{\Theta}_t,\mathbb{D}}(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t})$ a scalar disturbance bound that depends on the state and the input and takes into account both disturbance and parametric uncertainty and $L_{\overline{\Theta}_t}\delta_{k|t}$ a dynamic adjustment of the uncertainty bound based on the current tube size to capture the sensitivity of how much $\boldsymbol{G}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})$ changes as the perturbed state deviates from the nominal one. The scalar disturbance bound is further expressed in more detail as:

$$w_{\overline{\Theta}_t,\mathbb{D}}(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t}) = \eta_t c_{\mathbb{B}} \|\boldsymbol{G}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})\|_P + \bar{d} \tag{6.23}$$

with:

$$\bar{d} = max_{\boldsymbol{d}\in\mathbb{D}} \|\boldsymbol{d}\|_P = max_{\boldsymbol{d}\in\mathbb{D}} \sqrt{\boldsymbol{d}^\top \boldsymbol{P}\boldsymbol{d}} \tag{6.24}$$

the worst case disturbance effect, η_t the side of the hypercube described in (6.19), and $c_{\mathbb{B}} = \sqrt{n_\theta}$ where n_θ is the number of parameters in the unknown parameter vector $\boldsymbol{\theta}$. The second term of the sum in (6.22) which encodes a dynamic adjustment of the uncertainty bound based on the current tube size, depends on a dynamically adjustable, Lipschitz-like constant according to the size of the uncertainty set $\overline{\boldsymbol{\Theta}}_t$:

$$L_{\overline{\Theta}_t} = \eta_t L_{\mathbb{B}} \tag{6.25}$$

which describes the worst-case effect of the parametric uncertainty on the tube dynamics and it is dynamically adjusted based on the size of the hypercube η_t and scaled by another Liphchitz-like-constant:

$$L_{\mathbb{B}} = c_{\mathbb{B}} \max_{(\boldsymbol{x},\bar{\boldsymbol{x}},\bar{\boldsymbol{u}})\in\boldsymbol{\Psi}} \frac{\|\boldsymbol{G}(\bar{\boldsymbol{x}}, \boldsymbol{\kappa}(\boldsymbol{x}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})) - \boldsymbol{G}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})\|_P}{\|\boldsymbol{x} - \bar{\boldsymbol{x}}\|_P} \tag{6.26}$$

which captures the sensitivity of $\boldsymbol{G}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})$ with respect to the deviation of the perturbed state from the nominal one. The terms $\bar{d}$, $c_{\mathbb{B}}$, and $L_{\mathbb{B}}$ are computed offline while the Lipschitz-like constant in (6.25) is computed online based on the updated FPS. The worst-case effect of the mixed uncertainty (6.22) depends on the decision variables of the optimization problem (6.7) and thus it is evaluated in the optimization loop.

Having expressed the worst-case effect of the mixed uncertainty, we move on to the description of the tube dynamics. The size of this tube represents the worst deviation of the perturbed state from the nominal one due to the effect of the mixed uncertainty. It depends on the inherent stability properties of the system as well as the two sources of uncertainty, the disturbance and parametric uncertainty. The tube evolution is expressed by the following scalar, discrete dynamics:

$$\delta_{k+1|t} = \rho_{\bar{\boldsymbol{\theta}}_t}\delta_{k|t} + w_{k|t} \tag{6.27}$$

with $\rho_{\bar{\theta}_t}$ the *contraction rate* that depends on the incremental stabilizability property of the system, and $w_{k|t}$ the disturbance bound that acts as an excitation term on the tube size. While the disturbance bound is a decision variable in (6.7), the contraction rate $\rho_{\bar{\theta}_t}$ is updated online and stays fixed in the optimization loop:

$$\rho_{\bar{\theta}_t} = \rho_{\bar{\theta}_0} + (\eta_0 - \eta_t) L_{\mathbb{B},\rho} \tag{6.28}$$

with $\rho_{\bar{\theta}_0}$ derived from the contraction property of the incremental Lyapunov function, essentially a tuning parameter in the offline computations of the incremental Lyapunov function described in Section 6.3, and the Lipschitz-like term that bounds how much the system dynamics $G(\bar{x}, \bar{u})$ can change per deviation between the perturbed and nominal state accounting for the worst realization of the parametric uncertainty given as:

$$L_{\mathbb{B},\rho} = max_j \max_{(x,\bar{x},\bar{u})\in\Psi} \frac{\|G(\bar{x}, \kappa(x, \bar{x}, \bar{u})) - G(\bar{x}, \bar{u})\theta^j\|_P}{\|x - \bar{x}\|_P} \tag{6.29}$$

with $\theta^j \in \text{vert}(\overline{\Theta}_0),\ j = 1, \ldots, n_\theta$. The cascaded maximizations ensure that $L_{\mathbb{B},\rho}$ accounts for worst-case sensitivity within the feasible sets of states, inputs, and parameters. The terms $\rho_{\bar{\theta}_0}$, η_0, and $L_{\mathbb{B},\rho}$ are computed offline while $\rho_{\bar{\theta}_t}$ is updated online at each time step t. Then, (6.27) can be evaluated in the optimization loop to predict the evolution of the tube that will be used for constraint tightening as explained in Section 6.7.

For the contingency system, we follow the same approach as for the primary system, albeit simplified due to the consideration of the worst-case faulty parametric set $\underset{\sim}{\Theta}$ which is static and is not updated online. The total effect of the two types of uncertainties is denoted with $\underset{\sim}{w}_{\cdot|t}$, again a scalar decision variable in (6.7). We compute the worst-case realization of this mixed uncertainty to use as a lower bound in (6.7e) for the contingency plan:

$$w_{\underset{\sim}{\delta},\underset{\sim}{\Theta},\mathbb{D}}(\underset{\sim}{\bar{x}}_{k|t}, \underset{\sim}{\bar{u}}_{k|t}, \underset{\sim}{\delta}_{k|t}) = w_{\underset{\sim}{\Theta},\mathbb{D}}(\underset{\sim}{\bar{x}}_{k|t}, \underset{\sim}{\bar{u}}_{k|t}) + L_{\underset{\sim}{\Theta}} \underset{\sim}{\delta}_{k|t} \tag{6.30}$$

with:

$$w_{\Theta_t,\mathbb{D}}(\underset{\sim}{\bar{x}}_{k|t}, \underset{\sim}{\bar{u}}_{k|t}) = \underset{\sim}{\eta} c_{\mathbb{B}} \|G(\underset{\sim}{\bar{x}}, \underset{\sim}{\bar{u}})\|_P + \bar{d} \tag{6.31}$$

and:

$$L_{\underset{\sim}{\Theta}} = \underset{\sim}{\eta} L_{\mathbb{B}} \tag{6.32}$$

and where $\underset{\sim}{\eta}$ denotes the maximum size of the worst-case uncertainty set $\underset{\sim}{\Theta}$.

The tube evolution of the contingency plan is then expressed in the same manner by the following scalar, discrete dynamics:

$$\underset{\sim}{\delta}_{k+1|t} = \rho_{\underset{\sim}{\theta}} \underset{\sim}{\delta}_{k|t} + \underset{\sim}{w}_{k|t} \tag{6.33}$$

with the contraction rate $\rho_{\underset{\sim}{\theta}}$ computed as:

$$\rho_{\underset{\sim}{\theta}} = \rho_{\bar{\theta}_0} + (\eta_0 - \underset{\sim}{\eta}) L_{\mathbb{B},\rho} \tag{6.34}$$

6.7 Constraint Tightening

With the tube evolution established, we can utilize these results to tighten the nominal system and collision avoidance constraints presented in (6.5) and (6.6) respectively to

achieve robust constraint satisfaction. Following [202], the tightened constraints for the primary plan are expressed as:

$$\overline{\mathcal{Z}} := \overline{\mathcal{X}} \times \overline{\mathcal{U}} = \left\{ (\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}}) \in \mathbb{R}^{n_x \times n_u} \middle| \boldsymbol{h}^{\boldsymbol{s}}_{[j]}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}}) + \epsilon^{\boldsymbol{s}}_i \delta_{k|t} \leq 0, \quad j = 1, ..., n_{\boldsymbol{s}} \right\}, \tag{6.35}$$

and:

$$\overline{\mathcal{F}}_{\tau|t} := \left\{ \boldsymbol{p}_{k|t} \in \mathbb{R}^{n_p} \middle| \boldsymbol{h}^{\boldsymbol{o}}_{[j],k|t}(\boldsymbol{p}_{k|t}) + \epsilon^{\boldsymbol{o}} \delta_{k|t} \leq 0, \quad j = 1, ..., n_{\boldsymbol{o}} \right\}, \tag{6.36}$$

with $\overline{\mathcal{Z}} \subseteq \mathcal{Z}$ and $\overline{\mathcal{F}}_{\tau|t} \subseteq \mathcal{F}_{\tau|t}$ and the tightening constants computed based on incremental stabilizability ingredients of Section 6.3.

For the contingency plan, they are given in the same manner, although depending on the contingent tube evolution described in (6.33):

$$\widetilde{\mathcal{Z}} := \widetilde{\mathcal{X}} \times \widetilde{\mathcal{U}} = \left\{ (\underset{\sim}{\bar{\boldsymbol{x}}}, \underset{\sim}{\bar{\boldsymbol{u}}}) \in \mathbb{R}^{n_x \times n_u} \middle| \boldsymbol{h}^{\boldsymbol{s}}_{[j]}(\underset{\sim}{\bar{\boldsymbol{x}}}, \underset{\sim}{\bar{\boldsymbol{u}}}) + \epsilon^{\boldsymbol{s}}_i \underset{\sim}{\delta}_{k|t} \leq 0, \quad j = 1, ..., n_{\boldsymbol{s}} \right\}, \tag{6.37}$$

and:

$$\widetilde{\mathcal{F}}_{\tau|t} := \left\{ \boldsymbol{p}_{k|t} \in \mathbb{R}^{n_p} \middle| \boldsymbol{h}^{\boldsymbol{o}}_{[j],k|t}(\boldsymbol{p}_{k|t}) + \epsilon^{\boldsymbol{o}} \underset{\sim}{\delta}_{k|t} \leq 0, \quad j = 1, ..., n_{\boldsymbol{o}} \right\}, \tag{6.38}$$

with $\widetilde{\mathcal{Z}} \subseteq \overline{\mathcal{Z}} \subseteq \mathcal{Z}$ and $\widetilde{\mathcal{F}}_{\tau|t} \subseteq \overline{\mathcal{F}}_{\tau|t} \subseteq \mathcal{F}_{\tau|t}$.

A schematic illustration of this constraint tightening for the primary and contingency tightened collision avoidance constraints (6.36) and, designed as rule-compliant, half-spaces according to Chapter 3 is shown in Figure 6.3.

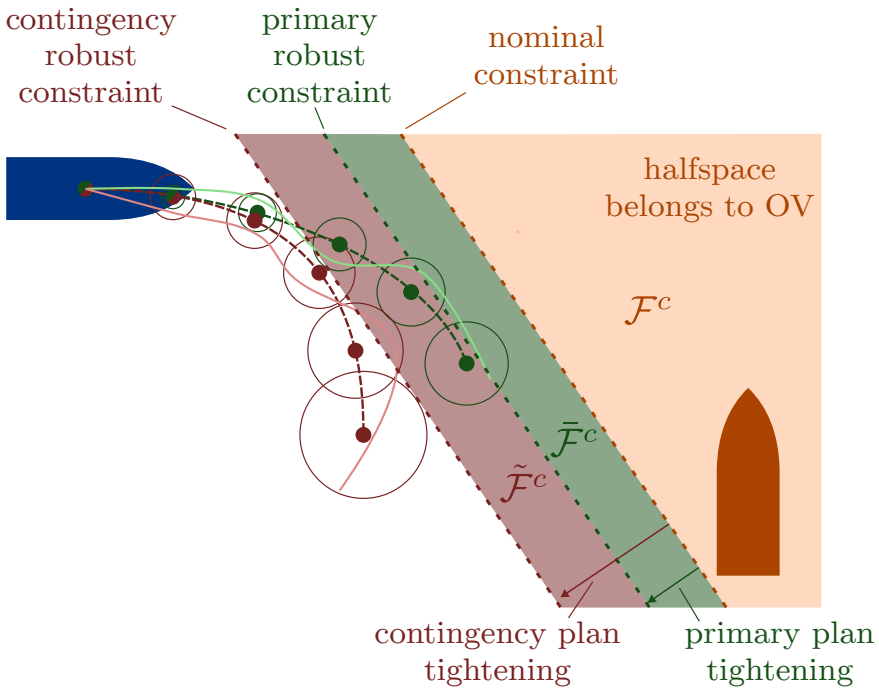

Figure 6.3: A schematic representation of how constraint tightening works for the rule-compliant collision avoidance constraints of Chapter 3. The colored areas represent the complements of the sets used in the optimization problem, i.e., the areas that the ASV is not allowed to enter.

6.8 Terminal Ingredients

For the primary plan, the terminal cost is then given as the following quadratic Lyapunov function:

$$V_f(\boldsymbol{x}, \bar{\boldsymbol{x}}) := V_\delta^2(\boldsymbol{x}, \bar{\boldsymbol{x}}) \frac{\alpha}{1 - \left(\rho_{\bar{\theta}_0} + L_{\overline{\Theta}_0}\right)^2} \tag{6.39}$$

while the terminal constraint set is defined as a sub-level set of the incremental Lyapunov function as:

$$\mathcal{X}_f = \left\{ (\boldsymbol{x}, \delta) \in \mathbb{R}^{n+1} \mid (V_\delta(\boldsymbol{x}, \bar{\boldsymbol{x}}) + \delta) \leq c_{x_\delta} \right\}, \tag{6.40}$$

where:

$$\rho_{\bar{\theta}_0} + L_{\overline{\Theta}_0} + c_{x_\delta} w_{\overline{\Theta}_t, \mathbb{D}}(\bar{\boldsymbol{x}}_{k|t}, \bar{\boldsymbol{u}}_{k|t}) \leq 1, \quad c_{x_\delta} = \min \left\{ -\boldsymbol{h}^{\boldsymbol{s}}(\boldsymbol{x}, \boldsymbol{u}) / \epsilon_i^{\boldsymbol{s}}, -\boldsymbol{h}_{k|t}^{\boldsymbol{o}}(\boldsymbol{p}_{k|t}) / \epsilon^{\boldsymbol{o}} \right\} \tag{6.41}$$

For the contingency plan, similar computations hold.

6.9 Overall Algorithm

The following two algorithms summarize the proposed offline design and the online operation. The main complexity in the offline design is the choice of a suitable function

Algorithm 6 Offline Algorithm

Input: $\boldsymbol{f_w}(\cdot)$, $\mathcal{Z}$, $\overline{\boldsymbol{\Theta}}_0$, $\underset{\sim}{\boldsymbol{\Theta}}$, $\rho_{\bar{\theta}_0}$, λ, $c_j^{\boldsymbol{s}}$, $c^{\boldsymbol{o}}$

Output: $\boldsymbol{P}$, $\boldsymbol{K}$, ${\epsilon_j}^{\boldsymbol{s}}$, $\epsilon^{\boldsymbol{o}}$, η_0, $\bar{d}$, $L_{\mathbb{B}}$, $L_{\mathbb{B},\rho}$, $c_{\mathbb{B}}$

1: Compute $\boldsymbol{P}, \boldsymbol{K}, {\epsilon_j}^{\boldsymbol{s}}, \epsilon^{\boldsymbol{o}}$ from (6.8), (6.9), (6.10)
2: $c_{\mathbb{B}} = \sqrt{n_{\boldsymbol{\theta}}}$
3: $\eta_0 = 0.5 \max_i \left(\boldsymbol{\theta}_{i,0}^{max} - \boldsymbol{\theta}_{i,0}^{min} \right)$
4: Compute $\bar{d}$ from (6.8)
5: Compute $L_{\mathbb{B}}$ from (6.26)
6: Compute $L_{\mathbb{B},\rho}$ from (6.29)

V_δ, solving the SDP (6.8) described in Section 6.3. The different constants that are used for the description of uncertainty can be computed similarly to a Lipschitz constant.

Algorithm 7 Online Algorithm

Input: $\boldsymbol{P}$, $\boldsymbol{K}$, $\epsilon_j{}^{\boldsymbol{s}}$, $\epsilon^{\boldsymbol{o}}$, η_0, $\bar{d}$, L_{B}, $L_{\mathrm{B},\rho}$, c_{B}
Output: $\boldsymbol{u}_t = \bar{\boldsymbol{u}}^*_{0|t} = \underset{\sim}{\bar{\boldsymbol{u}}}^*_{0|t}$

```
Compute predefined directions, 𝓔, from Algorithm 3
for t = 1,2,... do
    Get input-state data {x_t, x_{t-1}, u_t}
    Update collision avoidance constraints from Rule-Constraints Algorithm 1 Chapter 3
    Update FPS from FD Algorithm 5 from Chapter 5
    Update η_t = 0.5 max_i (θ^max_{i,t} − θ^min_{i,t}) and [θ̄_t]_i = (θ^min_{i,t} + θ^max_{i,t}) /2
    Update ρ_{θ̄_t} from (6.28)
    Update L_{Θ̄_t} from (6.25)
    Solve optimization problem (6.7)
    Apply control input u_t = ū*_{0|t} = ū*_{0|t}
end for
```

6.10 Conclusions

In this chapter, we presented a fault-tolerant trajectory optimization and control framework for uncertain nonlinear systems that are linear in the fault parameters and subject to additive state uncertainties as a response to Research Question **Q4**: "How can we jointly guarantee fault-tolerant and rule-compliant trajectories for ASVs operating in mixed-traffic environments?" This approach integrates the rule-compliant collision avoidance constraints developed in Chapter 3 with the fault diagnosis methodology from Chapter 5, which has the potential to enable fault-tolerant and rule-compliant trajectories among other traffic participants. The framework has the promise to achieve this by formulating both a primary and a contingency plan in a receding horizon manner. These plans are solved simultaneously, sharing the first control input of the input sequence. As a result, in the event of fault detection, a fail-safe trajectory is always available, ensuring safety under all circumstances. As such, the method could significantly improve the system's robustness and reliability, providing enhanced safety in dynamic and complex environments with multiple human-operated vehicles.

7

Conclusions & Future Work

This thesis investigates fault-tolerant motion planning and control in mixed-traffic environments, specifically applied to ASVs. Initially, traffic rules were integrated into an MPC-based trajectory optimization algorithm. This integration ensured rule compliance, dynamic feasibility, and scalability with respect to multiple OVs. The thesis further explores safety concerning faults through two distinct methods developed for FD of actuator faults. Finally, the MPC-based trajectory optimization algorithm was enhanced to handle these faults by leveraging diagnosis results and implementing reconfiguration in a RAMPC setting, thereby achieving FT.

This concluding chapter summarizes the thesis. Section 7.1 addresses the research questions posed in Section 1.2. Subsequently, Section 7.2 discusses existing limitations and proposes potential directions for future research.

7.1 Conclusions

In this section, we address the main research question and its associated sub-questions, summarizing the findings and contributions of this thesis.

7.1.1 Main Research Question

The main research question this thesis addressed is:

How can ASVs safely navigate in mixed traffic environments even in the presence of faults?

To comprehensively address this question, we delved into various critical aspects of fault-tolerant navigation within dense traffic environments. Our research findings present a comprehensive and integrated approach to ensuring the safety and reliability of ASVs. This is achieved through robust mechanisms for fault detection, isolation, and fault-tolerant motion planning, all while meticulously considering maritime traffic rules. The algorithms developed in this thesis predominantly rely on Model Predictive Control (MPC) due to its inherent flexibility and its capability to combine multiple control objectives. These objectives include collision avoidance, represented through state constraints, and fault tolerance, achieved by reconfiguring the control objectives, constraints, and even the system model itself. By leveraging MPC, we have successfully developed a versatile and resilient control strategy that addresses the complex requirements of autonomous navigation in mixed-traffic scenarios. The main research question has been thoroughly examined and answered through the detailed exploration of the following specific research questions.

7.1.2 Key Research Questions

Q1: *How can ASVs navigate safely and efficiently in dense traffic environments while ensuring compliance with maritime traffic rules?*

In Chapter 3, we developed an advanced MPC-based trajectory optimization algorithm that meticulously integrates maritime traffic rules as constraints. Utilizing the concepts of separating and supporting hyperplanes from optimization theory, we designed state constraints that permit only rule-compliant trajectories, while maintaining the convexity of the search space. This crucially ensures that ASVs adhere to these maritime traffic rules, facilitating not only compliance but also maintaining efficient navigation. Furthermore, the algorithm operates on simple state measurements, such as the position, heading, and velocities of other traffic participants, to assess the traffic situation through an FSM. This eliminates the necessity for an extensive communication framework to exchange plans and desired trajectories, making the algorithm highly suitable for scenarios involving conventional, non-autonomous traffic participants. Comprehensive simulations demonstrated that this approach effectively integrates rule compliance, dynamic feasibility, and scalability concerning multiple obstacle vessels (OVs). Consequently, it enables safe, efficient, and reliable navigation for ASVs operating in complex mixed-traffic environments.

Q2: *How to detect and isolate actuator faults in ASVs to enhance overall operational safety and reliability?*

Chapter 4 introduced a method for FD of actuator faults. This method utilized a combination of input-output data and a nonlinear observer to generate residuals, which were then coupled with adaptive thresholds designed to accurately detect the presence of faults. For the isolation of these faults, we capitalized on the inherent redundancy present in the actuation system, as well as the direct access to the control configuration provided by the MPC controller. This allowed us to effectively insulate specific states from particular control inputs, thereby isolating the faults more efficiently. The robustness and efficacy of this method were thoroughly validated through extensive simulation studies, which demonstrated its significant potential in enhancing the safety and reliability of ASV operations. The simulations confirmed that this approach not only detects faults with high accuracy but also ensures the continued safe operation of ASVs, thereby substantially contributing to their operational reliability in real-world scenarios.

Q3: *How can fault parameters be accurately and robustly estimated under varying operational conditions, including the presence of disturbances and noise?*

In Chapter 5, we concentrated on the robust estimation of fault parameters, an essential aspect of ensuring the reliability and safety of autonomous systems. By leveraging SME, we developed a sophisticated algorithm designed to accurately estimate fault parameters for nonlinear systems operating under a wide range of conditions, including varying operational scenarios, environmental disturbances, and measurement noise. This algorithm specifically estimates a feasible parameter set for the fault parameters, alongside a nominal estimate that resides within this feasible set. Fault detection was achieved through the implementation of inverse tests on the feasible parameter set, providing a rigorous method for identifying faults. The robustness and accuracy of this method were further validated by applying it to the same ASV model. This application demonstrated the method's effectiveness in fault detection and parameter estimation, even in the presence of significant state and output uncertainties in the operational environment. The results confirmed that our approach not only maintains high accuracy in fault estimation but also ensures reliable performance of ASVs under various challenging conditions, thereby enhancing their operational robustness and dependability.

Q4: *How can we jointly guarantee fault-tolerant and rule-compliant trajectories for ASVs operating in mixed-traffic environments?*

Chapter 6 seamlessly integrated the results of fault diagnosis into the MPC-based trajectory optimization algorithm, thereby establishing a comprehensive RAMPC framework. This advanced framework is designed to ensure that ASVs can dynamically reconfigure their trajectories in real-time response to detected faults, all while adhering to stringent maritime traffic rules. The integration allows for the automatic adjustment of the control strategy based on the real-time health status of the system, enhancing both safety and operational reliability. To fully validate the effectiveness of the RAMPC framework, further verification through extensive simulation studies is required. These evaluations will assess the framework's ability to maintain safe, efficient, and rule-compliant trajectories for ASVs operating in mixed-traffic environments, even in the presence of faults. Demonstrating its robustness in handling

faults while ensuring compliance with traffic regulations will be a crucial step toward confirming its practical applicability in real-world maritime operations.

7.2 Future Work

The challenge of ensuring safe, autonomous navigation in mixed-traffic environments remains a significant one. Although this thesis, along with other related works, has made strides in addressing this issue, there are still considerable challenges and areas for improvement before these algorithms can be deemed fully reliable and trustworthy for real-world applications and the safe deployment of autonomous systems. In the following sections, we highlight some of these limitations and propose research directions aimed at further enhancing the existing body of work.

1. *Global guidance planner for mixed-traffic environments*: To address the challenge of the local nature of the MPCC planner being potentially trapped in local minima, future work should focus on developing a higher-level guidance algorithm as in [203]. This algorithm could rely on a simpler system model and only accommodate collision checking to provide an initial trajectory guess that effectively navigates through the search space and avoids local minima. The guidance algorithm could incorporate search-based or sampling-based techniques to predict a globally near-optimal path that the MPCC planner can then refine. By leveraging such a higher-level strategy, the initial trajectory provided to the MPCC will significantly enhance the efficiency and success rate of the trajectory optimization process, particularly in complex and dynamic maritime environments. This approach will ensure that the MPCC planner starts from a more favorable point, thereby reducing computational effort and improving the overall robustness of the navigation solution.

2. *Convexification of the trajectory optimizer*: Future work should focus on the convexification of the entire trajectory optimization process within the MPCC framework. This can be achieved by linearizing the system dynamics and the contouring and lag error terms in the objective function across the prediction horizon. By accounting for the linearization error (higher order terms in Taylor expansion) and robustifying against it, we can follow the initial trajectory provided by the guidance algorithm and then optimize it to obtain smooth, dynamically feasible trajectories concerning multiple obstacles. This approach simplifies the optimization problem, allowing it to be solved in real-time using dedicated solvers for example OSQP [179] and Tiny-MPC [204]. The convexification process not only improves computational efficiency but also enhances the feasibility and robustness of the optimized trajectories, ensuring reliable navigation in complex, dynamic environments.

3. *Softening collision avoidance constraints with slack variables and lexicographic optimization*: This way we can still have a formal description of the collision avoidance constraints without relying on the heuristic tuning of the objective function but at the same time resolve feasibility issues. Slack variables will convert the hard constraints to soft thus minimizing the risk for infeasibility. Lexicographic optimization can omit the collision avoidance constraints if the solver is unable to find a collision.

4. *Robustness testing with rules-related parameter variation and comparison with communication schemes*: To further validate and enhance the robustness of the proposed algorithm, future research should involve applying the same MPCC algorithm with the introduction of some variation in the parameters related to the maritime traffic rules. This study should include multiple ASVs navigating among each other without communication, thereby simulating more realistic operational conditions. The performance of the ASVs can be compared with distributed coordination algorithms [205, 154], which necessitates communication between vessels. Key metrics for comparison should include the optimality of the chosen trajectories, the time elapsed to reach the destination and the overall success rate of navigation. This comparative analysis will provide deeper insights into the advantages and limitations of communication-free navigation strategies and highlight potential areas for improvement in autonomous vessel coordination.

5. *Nonlinear adaptive observer for fault parameter estimation*: Future work should extend the current observer to a nonlinear adaptive observer capable of estimating fault parameters as well. This approach should be compared with SME to evaluate their respective strengths and weaknesses. It is critical to highlight the differences between fault detection FD in the state space, which uses residuals and thresholds, and fault detection in the parameter space, where we have explicit access to the parameters. By implementing and comparing these methods, we can gain a deeper understanding of their effectiveness. The goal is to identify potential complementarities and synergies between these methods, enabling the development of more robust and efficient fault diagnosis techniques. Such an in-depth comparison will provide valuable insights into the strengths and limitations of each method, guiding future enhancements in the field of autonomous system fault diagnosis.

6. *Generalization to broader system models for SME*: Future research should aim to extend the methodology developed for fault detection and estimation with SME to more general nonlinear system models. In this thesis, we considered nonlinear systems represented as $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{p})$ that can be written as $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \tilde{\boldsymbol{g}}(\boldsymbol{u})\boldsymbol{p}$ (linear to the parameters). The method needs to be extended to more general systems in the non-separable form $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p})$ that can be written as $\dot{\boldsymbol{x}} = \tilde{\boldsymbol{f}}(\boldsymbol{x}, \boldsymbol{u})\boldsymbol{p}$, while still utilizing interval arithmetic tools. The primary condition is that the system must remain linear with respect to the parameters of interest. By expanding the applicability of these methods to a wider range of system models, we can enhance the versatility of the FD method, making it applicable to a broader array of practical autonomous systems.

7. *Comparative analysis of FPS approximations and regularization methods*: Future work should include an extensive comparison of different approximations of the FPS and various regularization processes. By systematically exploring and tuning polytope approximations and regularization techniques, we can significantly improve the accuracy and robustness of fault parameter estimation. This comparative analysis should focus on identifying the strengths and weaknesses of each approach, understanding how they perform under different operational conditions, and determining their applicability to various autonomous system models. Such detailed comparisons

will provide valuable insights into the most effective strategies for enhancing the reliability and performance of SME, ultimately contributing to safer and more efficient ASV operations.

8. *Extension of FD methods to include sensor faults*: While this thesis primarily addresses actuator faults, which are closely related to the equations of motion of the ASV, sensor faults have not been considered. Sensor faults are critically important to the safety of ASVs as they directly impact the perception of the surrounding traffic environment. Future work should include an extensive analysis of sensor faults, incorporating robustness to disturbances, noise, and faults in sensor readings. By applying fault detection and diagnosis methods to sensors, we can significantly enhance the overall safety and reliability of ASVs, ensuring robust and fault-tolerant navigation even in the presence of sensor anomalies.

9. *Validation of the method proposed in Chapter 6*: The method presented in Chapter 6 demonstrates significant potential for addressing the challenge of rule-compliant and fault-tolerant navigation in environments with human-operated vehicles. However, its effectiveness remains unverified due to the absence of simulation results. To ensure comprehensive evaluation, the framework should be implemented within the existing codebase and validated through simulation experiments, following the approach employed in previous chapters.

10. *On Assumptions and Robustness.* The methods developed in this thesis are based on several modelling and operational assumptions—such as bounded and uncorrelated noise, instantaneous actuator faults, always-feasible collision avoidance, and accurate hydrodynamic models—which were adopted to enable focused algorithmic development. However, a systematic investigation into how violations of these assumptions affect performance is warranted. For example, environmental disturbances such as wind, waves, and currents may display complex dynamics or interactions that are not adequately captured by fixed, bounded models. Additionally, sensor faults were not considered in this work, but their presence in practice could significantly complicate fault detection and hinder the ability to distinguish between different types of faults. Furthermore, operation in confined waters or the risk of grounding would necessitate fail-safe re-planning strategies capable of handling infeasibility within the MPC framework. Addressing these challenges will require both theoretical robustness analysis and extensive validation in high-fidelity simulators or sea trials under diverse conditions. Bridging these gaps constitutes a crucial direction for future work and is essential for transitioning the proposed methods toward reliable and certifiable autonomous vessel operation.

Bibliography

[1] Our World In Data. *Causes of Death*. https://ourworldindata.org/causes-of-death. Accessed: 2022-10-30.

[2] ASIRT. *Road Safety Facts*. https://www.asirt.org/facts/. Accessed: 2022-10-30.

[3] Santokh Singh. "Critical reasons for crashes investigated in the National Motor Vehicle Crash Causation Survey". In: *Traffic Safety Facts Crash Stats, Report No. DOT HS 812 506* (2018).

[4] Cruise. *The Cruise Origin Story*. Accessed: 2024-10-30. 2020. URL: https://www.getcruise.com/news/blog/2020/the-cruise-origin-story/.

[5] Yara. *Yara Birkeland Press Kit*. Accessed: 2024-10-30. 2024. URL: https://www.yara.com/news-and-media/media-library/press-kits/yara-birkeland-press-kit/.

[6] EMSA. *Annual Overview of Marine Casualties and Incidents*. http://www.emsa.europa.eu. Accessed: 2024-10-30.

[7] Saildrone. *Ocean Research Solutions*. Accessed: 2024-10-30. 2024. URL: https://www.saildrone.com/solutions/ocean-research.

[8] Maritime Robotics. *Otter USV*. Accessed: 2024-10-30. 2024. URL: https://www.maritimerobotics.com/otter.

[9] Martin Stopford. *Maritime Economics*. Taylor and Francis, 2008. ISBN: 9781134476527.

[10] Linying Chen. "Cooperative Multi-Vessel Systems for Waterborne Transport". PhD thesis. Delft University of Technology, 2019.

[11] CNBC. *Cruise Traffic Jam After California Approves 24/7 Robotaxi Service*. Accessed: 2024-10-30. URL: https://www.cnbc.com/2023/08/14/cruise-traffic-jam-after-california-approves-24-7-robotaxi-service.html.

[12] Johannes Moscherosch. *Collaboration to Leverage AI and Analytics for Vessel Traffic Management*. Accessed: 2024-10-30. 2024. URL: https://www.linkedin.com/pulse/collaboration-leverage-ai-analytics-vessel-traffic-port-moscherosch/.

[13] National Transportation Safety Board. *Collision Between Vehicle Controlled by Developmental Automated Driving System and Pedestrian, Tempe, Arizona, March 18, 2018*. Highway Accident Report. Accessed: 2022-10-30. Washington, DC: National Transportation Safety Board, 2019.

[14] National Highway Traffic Safety Administration. *Tesla Recalls Over 40,000 Vehicles for Possible Loss of Power Steering*. Tech. rep. 22V-818. Accessed: 2022-10-30. National Highway Traffic Safety Administration, Nov. 2022.

[15] National Transportation Safety Board. *Collision Between US Navy Destroyer Fitzgerald and Philippine-Flag Container Ship ACX Crystal*. Marine Accident Report. Accessed: 2022-10-30. National Transportation Safety Board, 2020.

[16] National Transportation Safety Board. *Investigation into the Collision of a Container Ship with Baltimore Bridge*. Preliminary Report. Expected publication year: 2024. Check NTSB website for updates and final report. National Transportation Safety Board, 2023.

[17] The Verge. *Uber at Fault in Self-Driving Crash, Says NTSB*. Accessed: 2024-30-10. 2019. URL: https://www.theverge.com/2019/11/19/20972584/uber-fault-self-driving-crash-ntsb-probable-cause.

[18] Wired. *Baltimore Bridge Collapse Disrupts Shipping Supply Chain*. Accessed: 2024-30-10. 2024. URL: https://www.wired.com/story/baltimore-bridge-collapse-shipping-supply-chain-disruption-francis-scott-key/.

[19] Basil Kouvaritakis and Mark Cannon. "Model Predictive Control". In: *Switzerland: Springer International Publishing* 38 (2016), pp. 13–56.

[20] James Blake Rawlings, David Q Mayne, Moritz Diehl, et al. *Model Predictive Control: Theory, Computation, and Design*. Vol. 2. Nob Hill Publishing Madison, WI, 2017.

[21] A. Tsolakis and R. R. Negenborn and V. Reppa and L. Ferranti. "Model Predictive Trajectory Optimization and Control for Autonomous Surface Vessels Considering Traffic Rules". In: *IEEE Transactions on Intelligent Transportation Systems* (2024).

[22] Anastasios Tsolakis, Laura Ferranti, and Vasso Reppa. "Active Thruster Fault Diagnosis for an Overactuated Autonomous Surface Vessel". In: *IFAC-PapersOnLine* 1.1 (2024), p. 1.

[23] A. Tsolakis, L. Ferranti, and V. Reppa. "Set-Membership for Fault Diagnosis of Nonlinear Systems". In: *IEEE European Control Conference* (2025).

[24] Roland Siegwart, Illah R. Nourbakhsh, and Davide Scaramuzza. *Introduction to Autonomous Mobile Robots*. 2nd. The MIT Press, 2011. ISBN: 0262015358.

[25] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. "A survey of motion planning and control techniques for self-driving urban vehicles". In: *IEEE Transactions on Intelligent Vehicles* 1.1 (2016), pp. 33–55.

[26] Steven M. LaValle. *Planning Algorithms*. USA: Cambridge University Press, 2006. ISBN: 0521862051.

[27] John Hershberger. "An optimal visibility graph algorithm for triangulated simple polygons". In: *Algorithmica* 4.1 (1989), pp. 141–155.

[28] Han-Pang Huang and Shu-Yun Chung. "Dynamic visibility graph for path planning". In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*. Vol. 3. IEEE. 2004, pp. 2813–2818.

[29] Rahul Kala. "Rapidly exploring random graphs: motion planning of multiple mobile robots". In: *Advanced Robotics* 27.14 (2013), pp. 1113–1122.

[30] Edsger W Dijkstra. "A note on two problems in connexion with graphs". In: *Numerische mathematik* 1.1 (1959), pp. 269–271.

[31] Peter E Hart, Nils J Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[32] Anthony Stentz et al. "The focussed dˆ* algorithm for real-time replanning". In: *IJCAI*. Vol. 95. 1995, pp. 1652–1659.

[33] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In: *IEEE transactions on Robotics and Automation* 12.4 (1996), pp. 566–580.

[34] James J Kuffner and Steven M LaValle. "RRT-connect: An efficient approach to single-query path planning". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*. Vol. 2. IEEE. 2000, pp. 995–1001.

[35] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*. Vol. 56. Springer, 2009.

[36] Georges S Aoude, Brandon D Luders, Joshua M Joseph, Nicholas Roy, and Jonathan P How. "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns". In: *Autonomous Robots* 35.1 (2013), pp. 51–76.

[37] Hai Zhu and Javier Alonso-Mora. "Chance-constrained collision avoidance for mavs in dynamic environments". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 776–783.

[38] Tirthankar Bandyopadhyay, Kok Sung Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. "Intention-aware motion planning". In: *Algorithmic foundations of robotics X*. Springer, 2013, pp. 475–491.

[39] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. "Probabilistic decision-making under uncertainty for autonomous driving using continuous POMDPs". In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2014, pp. 392–399.

[40] Bingyu Zhou, Wilko Schwarting, Daniela Rus, and Javier Alonso-Mora. "Joint multi-policy behavior estimation and receding-horizon trajectory planning for automated urban driving". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2388–2394.

[41] Constantin Hubmann, Jens Schulz, Marvin Becker, Daniel Althoff, and Christoph Stiller. "Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction". In: *IEEE transactions on intelligent vehicles* 3.1 (2018), pp. 5–17.

[42] Xin Huang, Sungkweon Hong, Andreas Hofmann, and Brian C Williams. "Online risk-bounded motion planning for autonomous vehicles in dynamic environments". In: *Proceedings of the International Conference on Automated Planning and Scheduling*. Vol. 29. 2019, pp. 214–222.

[43] Yuanzhe Wang, Danwei Wang, and Senqiang Zhu. "A new navigation function based decentralized control of multi-vehicle systems in unknown environments". In: *Journal of Intelligent & Robotic Systems* 87.2 (2017), pp. 363–377.

[44] Annemarie Turnwald and Dirk Wollherr. "Human-like motion planning based on game theoretic decision making". In: *International Journal of Social Robotics* 11.1 (2019), pp. 151–170.

[45] Jaime F Fisac, Eli Bronstein, Elis Stefansson, Dorsa Sadigh, S Shankar Sastry, and Anca D Dragan. "Hierarchical game-theoretic planning for autonomous vehicles". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 9590–9596.

[46] Frank Havlak and Mark Campbell. "Discrete and continuous, probabilistic anticipation for autonomous robots in urban environments". In: *IEEE Transactions on Robotics* 30.2 (2013), pp. 461–474.

[47] Quan Tran and Jonas Firl. "Modelling of traffic situations at urban intersections with probabilistic non-parametric regression". In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2013, pp. 334–339.

[48] Yu Fan Chen, Michael Everett, Miao Liu, and Jonathan P How. "Socially aware motion planning with deep reinforcement learning". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 1343–1350.

[49] Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P How. "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 285–292.

[50] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer-Verlag, 2007. ISBN: 354023957X.

[51] Oussama Khatib. "Real-time obstacle avoidance for manipulators and mobile robots". In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation*. Vol. 2. IEEE. 1985, pp. 500–505.

[52] Rekha Raja, Ashish Dutta, and KS Venkatesh. "New potential field method for rough terrain path planning using genetic algorithm for a 6-wheel rover". In: *Robotics and Autonomous Systems* 72 (2015), pp. 295–306.

[53] Shuzhi Sam Ge and Yun J Cui. "Dynamic motion planning for mobile robots using potential field method". In: *Autonomous Robots* 13.3 (2002), pp. 207–222.

[54] Bakir Lacevic and Paolo Rocco. "Kinetostatic danger field - a novel safety assessment for human-robot interaction". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2010, pp. 2169–2174.

[55] Elon Rimon and Daniel E Koditschek. "Exact robot navigation using artificial potential functions". In: *Departmental Papers (ESE)* (1992), p. 323.

[56] Z Kan, AP Dani, JM Shea, and WE Dixon. "Ensuring network connectivity during formation control using a decentralized navigation function". In: *2010-MILCOM 2010 Military Communications Conference*. IEEE. 2010, pp. 531–536.

[57] Johann Borenstein, Yoram Koren, et al. "The vector field histogram-fast obstacle avoidance for mobile robots". In: *IEEE Transactions on Robotics and Automation* 7.3 (1991), pp. 278–288.

[58] Javier Minguez. "The obstacle-restriction method for robot obstacle avoidance in difficult environments". In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2005, pp. 2284–2290.

[59] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. "The dynamic window approach to collision avoidance". In: *IEEE Robotics & Automation Magazine* 4.1 (1997), pp. 23–33.

[60] Paolo Fiorini and Zvi Shiller. "Motion planning in dynamic environments using velocity obstacles". In: *The International Journal of Robotics Research* 17.7 (1998), pp. 760–772.

[61] David Wilkie, Jur Van Den Berg, and Dinesh Manocha. "Generalized velocity obstacles". In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2009, pp. 5573–5578.

[62] Jur Van den Berg, Ming Lin, and Dinesh Manocha. "Reciprocal velocity obstacles for real-time multi-agent navigation". In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 1928–1935.

[63] Jur Van Den Berg, Jamie Snape, Stephen J Guy, and Dinesh Manocha. "Reciprocal collision avoidance with acceleration-velocity obstacles". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3475–3482.

[64] Jur Van Den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. "Reciprocal n-body collision avoidance". In: *Robotics Research*. Springer, 2011, pp. 3–19.

[65] Alexander Liniger, Alexander Domahidi, and Manfred Morari. "Optimization-based autonomous racing of 1:43 scale RC cars". In: *Optimal Control Applications and Methods* 36.5 (2015), pp. 628–647.

[66] Wilko Schwarting, Javier Alonso-Mora, Liam Paull, Sertac Karaman, and Daniela Rus. "Safe Nonlinear Trajectory Generation for Parallel Autonomy with a Dynamic Vehicle Model". In: *IEEE Transactions on Intelligent Transportation Systems* 19.9 (2018), pp. 2994–3008.

[67] Bruno Brito, Boaz Floor, Laura Ferranti, and Javier Alonso-Mora. "Model Predictive Contouring Control for Collision Avoidance in Unstructured Dynamic Environments". In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4459–4466.

[68] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. "Aggressive driving with model predictive path integral control". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 1433–1440.

[69] Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. "Model predictive path integral control: From theory to parallel computation". In: *Journal of Guidance, Control, and Dynamics* 40.2 (2017), pp. 344–357.

[70] Manan S Gandhi, Bogdan Vlahov, Jason Gibson, Grady Williams, and Evangelos A Theodorou. "Robust model predictive path integral control: Analysis and performance guarantees". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1423–1430.

[71] Zhixiang Liu, Youmin Zhang, Xiang Yu, and Chi Yuan. "Unmanned surface vehicles: An overview of developments and challenges". In: *Annual Reviews in Control* 41 (2016), pp. 71–93.

[72] Chongfeng Wei, Richard Romano, Natasha Merat, Yafei Wang, Chuan Hu, Hamid Taghavifar, Foroogh Hajiseyedjavadi, and Erwin R Boer. "Risk-based autonomous vehicle motion control with considering human driver's behaviour". In: *Transportation research part C: emerging technologies* 107 (2019), pp. 1–14.

[73] Christian Pek, Stefanie Manzinger, Markus Koschi, and Matthias Althoff. "Using online verification to prevent autonomous vehicles from causing accidents". In: *Nature Machine Intelligence* 2.9 (2020), pp. 518–528.

[74] Albert Rizaldi and Matthias Althoff. "Formalising traffic rules for accountability of autonomous vehicles". In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE. 2015, pp. 1658–1665.

[75] Markus Koschi, Christian Pek, Mona Beikirch, and Matthias Althoff. "Set-based prediction of pedestrians in urban environments considering formalized traffic rules". In: *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE. 2018, pp. 2704–2711.

[76] Christian Pek, Peter Zahn, and Matthias Althoff. "Verifying the safety of lane change maneuvers of self-driving vehicles based on formalized traffic rules". In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2017, pp. 1477–1483.

[77] Maximilian Geisslinger, Franziska Poszler, and Markus Lienkamp. "An ethical trajectory planning algorithm for autonomous vehicles". In: *Nature Machine Intelligence* 5.2 (2023), pp. 137–144.

[78] Karena X Cai, Tung Phan-Minh, Soon-Jo Chung, and Richard M Murray. "Rules of the Road: Formal Guarantees for Autonomous Vehicles With Behavioral Contract Design". In: *IEEE Transactions on Robotics* (2023).

[79] Jesper Karlsson and Jana Tumova. "Intention-aware motion planning with road rules". In: *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*. IEEE. 2020, pp. 526–532.

[80] Freddy A Boulton, Elena Corina Grigore, and Eric M Wolff. "Motion prediction using trajectory sets and self-driving domain knowledge". In: *arXiv preprint arXiv:2006.04767* (2020).

[81] Yamin Huang, Linying Chen, Pengfei Chen, Rudy R Negenborn, and P.H.A.J.M. van Gelder. "Ship collision avoidance methods". In: *Safety Science* 121 (2020), pp. 451–473.

[82] "COLREGs - International Regulations for Preventing Collisions at Sea". In: *Convention on the International Regulations for Preventing Collisions at Sea, 1972* (1972), pp. 1–74.

[83] Azzeddine Bakdi and Erik Vanem. "Fullest COLREGs evaluation using fuzzy logic for collaborative decision-making analysis of autonomous ships in complex situations". In: *IEEE Transactions on Intelligent Transportation Systems* 23.10 (2022), pp. 18433–18445.

[84] Yonghoon Cho, Jungwook Han, and Jinwhan Kim. "Intent inference of ship maneuvering for automatic ship collision avoidance". In: *IFAC-PapersOnLine* 51.29 (2018), pp. 384–388.

[85] Sverre Velten Rothmund, Trym Tengesdal, Edmund Førland Brekke, and Tor Arne Johansen. "Intention modeling and inference for autonomous collision avoidance at sea". In: *Ocean Engineering* 266 (2022), p. 113080.

[86] Peter Nicholas Hansen, Dimitrios Papageorgiou, Mogens Blanke, Roberto Galeazzi, Marie Lützen, John Mogensen, Mette Bennedsen, and Dorte Hansen. "COLREGs-based situation awareness for marine vessels-a discrete event systems approach". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 14501–14508.

[87] Yoshiaki Kuwata, Michael T Wolf, Dimitri Zarzhitsky, and Terrance L Huntsberger. "Safe maritime autonomous navigation with COLREGs, using velocity obstacles". In: *IEEE Journal of Oceanic Engineering* 39.1 (2013), pp. 110–119.

[88] Emil H Thyri and Morten Breivik. "Partly COLREGs-compliant collision avoidance for ASVs using encounter-specific velocity obstacles". In: *IFAC-PapersOnLine* 55.31 (2022), pp. 37–43.

[89] Yonghoon Cho, Jungwook Han, and Jinwhan Kim. "Efficient COLREG-Compliant Collision Avoidance in Multi-Ship Encounter Situations". In: *IEEE Transactions on Intelligent Transportation Systems* (2020), pp. 1–13.

[90] D. K.M. Kufoalor, E. F. Brekke, and T. A. Johansen. "Proactive Collision Avoidance for ASVs using A Dynamic Reciprocal Velocity Obstacles Method". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 2402–2409.

[91] Yuxin Zhao, Wang Li, and Peng Shi. "A real-time collision avoidance learning system for Unmanned Surface Vessels". In: *Neurocomputing* 182 (2016), pp. 255–266.

[92] Y Xue, BS Lee, and D Han. "Automatic collision avoidance of ships". In: *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment* 223.1 (2009), pp. 33–46.

[93] Wasif Naeem, Sable C. Henrique, and Liang Hu. "A Reactive COLREGs-Compliant Navigation Strategy for Autonomous Maritime Navigation". In: *IFAC-PapersOnLine* 49.23 (2016), pp. 207–213. ISSN: 24058963.

[94] Hongguang Lyu and Yong Yin. "COLREGS-Constrained Real-Time Path Planning for Autonomous Ships Using Modified Artificial Potential Fields". In: *Journal of Navigation* 72.3 (2019), pp. 588–608.

[95] Petr Švec, Atul Thakur, Eric Raboin, Brual C Shah, and Satyandra K Gupta. "Target following with motion prediction for unmanned surface vehicle operating in cluttered environments". In: *Autonomous Robots* 36.4 (2014), pp. 383–405.

[96] Pranay Agrawal and John M. Dolan. "COLREGS-compliant target following for an Unmanned Surface Vehicle in dynamic environments". In: *IEEE International Conference on Intelligent Robots and Systems* 2015-Decem (2015), pp. 1065–1070.

[97] Zhibo He, Chenguang Liu, Xiumin Chu, Rudy R Negenborn, and Qing Wu. "Dynamic anti-collision A-star algorithm for multi-ship encounter situations". In: *Applied Ocean Research* 118 (2022), p. 102995.

[98] Mauro Candeloro, Anastasios M Lekkas, and Asgeir J Sørensen. "A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels". In: *Controlunder Engineering Practice* 61 (2017), pp. 41–54.

[99] Hao Tien Lewis Chiang and Lydia Tapia. "COLREG-RRT: An RRT-Based COLREGS-Compliant Motion Planner for Surface Vehicle Navigation". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 2024–2031.

[100] Thomas Thuesen Enevoldsen, Christopher Reinartz, and Roberto Galeazzi. "COLREGs-Informed RRT* for Collision Avoidance of Marine Crafts". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 8083–8089.

[101] Eivind Meyer, Amalie Heiberg, Adil Rasheed, and Omer San. "COLREG-compliant collision avoidance for unmanned surface vehicle using deep reinforcement learning". In: *IEEE Access* 8 (2020), pp. 165344–165364.

[102] Xinli Xu, Yu Lu, Xiaocheng Liu, and Weidong Zhang. "Intelligent collision avoidance algorithms for USVs via deep reinforcement learning under COLREGs". In: *Ocean Engineering* 217 (2020), p. 107704.

[103] Tor Arne Johansen, Tristan Perez, and Andrea Cristofaro. "Ship collision avoidance and COLREGS compliance using simulation-based control behavior selection with predictive hazard assessment". In: *IEEE transactions on intelligent transportation systems* 17.12 (2016), pp. 3407–3422.

[104] Inger Berge Hagen, D Kwame Minde Kufoalor, Edmund Førland Brekke, and Tor Arne Johansen. "MPC-based collision avoidance strategy for existing marine vessel guidance systems". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 7618–7623.

[105] D. K.M. Kufoalor, E. Wilthil, I. B. Hagen, E. F. Brekke, and T. A. Johansen. "Autonomous colregs-compliant decision making using maritime radar tracking and model predictive control". In: *2019 18th European Control Conference, ECC 2019* (2019), pp. 2536–2542.

[106] Trym Tengesdal, Tor A Johansen, and Edmund F Brekke. "Ship collision avoidance utilizing the cross-entropy method for collision risk assessment". In: *IEEE Transactions on Intelligent Transportation Systems* 23.8 (2021), pp. 11148–11161.

[107] IB Hagen, DKM Kufoalor, TA Johansen, and EF Brekke. "Scenario-Based Model Predictive Control with Several Steps for COLREGs Compliant Ship Collision Avoidance". In: *IFAC-PapersOnLine* 55.31 (2022), pp. 307–312.

[108] Liang Hu, Wasif Naeem, Eshan Rajabally, Graham Watson, Terry Mills, Zakirul Bhuiyan, Craig Raeburn, Ivor Salter, and Claire Pekcan. "A multiobjective optimization approach for COLREGs-compliant path planning of autonomous surface vehicles verified on networked bridge simulators". In: *IEEE Transactions on Intelligent Transportation Systems* 21.3 (2019), pp. 1167–1179.

[109] Mohamed Abdelaal, Martin Fränzle, and Axel Hahn. "Nonlinear Model Predictive Control for trajectory tracking and collision avoidance of underactuated vessels with disturbances". In: *Ocean Engineering* 160.April (2018), pp. 168–180.

[110] Bjørn Olav H. Eriksen, Glenn Bitar, Morten Breivik, and Anastasios M. Lekkas. "Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 13–17". In: *Frontiers in Robotics and AI* 7.February (2020), pp. 1–18.

[111] Zhe Du, Rudy R. Negenborn, and Vasso Reppa. "Multi-Objective Cooperative Control for a Ship-Towing System in Congested Water Traffic Environments". In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022), pp. 24318–24329.

[112] Jitske de Vries, Elia Trevisan, Jules van der Toorn, Tuhin Das, Bruno Brito, and Javier Alonso-Mora. "Regulations Aware Motion Planning for Autonomous Surface Vessels in Urban Canals". In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 3291–3297.

[113] Mohamed Abdelaal and Axel Hahn. "NMPC-based trajectory tracking and collision avoidance of unmanned surface vessels with rule-based COLREGs confinement". In: *2016 IEEE Conference on Systems, Process and Control (ICSPC)*. IEEE. 2016, pp. 23–28.

[114] Emil H. Thyri and Morten Breivik. "Collision avoidance for ASVs through trajectory planning: MPC with COLREGs-compliant nonlinear constraints". In: *Modeling, Identification and Control* 43.2 (2022), pp. 55–77.

[115] Yashar Shabbouei Hagh, Reza Mohammadi Asl, Afef Fekih, Huapeng Wu, and Heikki Handroos. "Active fault-tolerant control design for actuator fault mitigation in robotic manipulators". In: *IEEE Access* 9 (2021), pp. 47912–47929.

[116] Wenjie Zhang, Xiaohui Yang, Zhenghong Xu, Wei Zhang, Li Yang, and Xiaoping Liu. "An adaptive fault-tolerant control method for robot manipulators". In: *International Journal of Control, Automation and Systems* 19.12 (2021), pp. 3983–3995.

[117] Mien Van, Michalis Mavrovouniotis, and Shuzhi Sam Ge. "An adaptive backstepping nonsingular fast terminal sliding mode control for robust fault tolerant control of robot manipulators". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 49.7 (2018), pp. 1448–1458.

[118] Mohamed A Kamel, Xiang Yu, and Youmin Zhang. "Fault-tolerant cooperative control design of multiple wheeled mobile robots". In: *IEEE Transactions on control systems technology* 26.2 (2017), pp. 756–764.

[119] George C. Karras and George K. Fourlas. "Model Predictive Fault Tolerant Control for Omni-directional Mobile Robots". In: *Journal of Intelligent and Robotic Systems* 97 (2020), pp. 635–655.

[120] Andrea Cristofaro and Tor Arne Johansen. "Fault tolerant control allocation using unknown input observers". In: *Automatica* 50.7 (2014), pp. 1891–1897.

[121] Qingrui Zhang, Xinyu Zhang, Bo Zhu, and Vasso Reppa. "Fault Tolerant Control for Autonomous Surface Vehicles Via Model Reference Reinforcement Learning". In: *Proceedings of the 60th IEEE Conference on Decision and Control (CDC)*. IEEE. 2021, pp. 1536–1541.

[122] Alessandro Baldini, Riccardo Felicetti, Alessandro Freddi, Sauro Longhi, and Andrea Monteriù. "Fault Tolerant Control for Remotely Operated Vehicles with Thruster Faults Using Nonlinear Disturbance Observers". In: *IFAC-PapersOnLine* 55.31 (2022), pp. 275–280.

[123] Hojjat A. Izadi, Youmin Zhang, and Brandon W. Gordon. "Fault Tolerant Model Predictive Control of Quad-rotor Helicopters with Actuator Fault Estimation". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 6343–6348.

[124] Mark W. Mueller and Raffaello D'Andrea. "Stability and Control of a Quadrocopter Despite the Complete Loss of One, Two, or Three Propellers". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 45–52.

[125] Fang Nan, Sihao Sun, Philipp Foehn, and Davide Scaramuzza. "Nonlinear MPC for Quadrotor Fault-Tolerant Control". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 5047–5054.

[126] Meng Guo, Dimos V. Dimarogonas, and Karl Henrik Johansson. "Distributed Real-Time Fault Detection and Isolation for Cooperative Multi-Agent Systems". In: *Proceedings of the American Control Conference (ACC)*. IEEE. 2012, pp. 5270–5275.

[127] Eliahu Khalastchi and Meir Kalech. "On fault detection and diagnosis in robotic systems". In: *ACM Computing Surveys (CSUR)* 51.1 (2018), pp. 1–24.

[128] Feng Xu, Vicenç Puig, Carlos Ocampo-Martinez, Sorin Olaru, and Silviu-Iulian Niculescu. "Robust MPC for actuator-fault tolerance using set-based passive fault detection and active fault isolation". In: *53rd IEEE Conference on Decision and Control*. IEEE. 2014, pp. 4959–4964.

[129] Feng Xu, Vicenç Puig, Carlos Ocampo-Martinez, Sorin Olaru, and Florin Stoican. "Set-theoretic methods in robust detection and isolation of sensor faults". In: *International Journal of Systems Science* 46.13 (2015), pp. 2317–2334.

[130] Joseph K Scott, Davide M Raimondo, Giuseppe Roberto Marseglia, and Richard D Braatz. "Constrained zonotopes: A new tool for set-based estimation and fault detection". In: *Automatica* 69 (2016), pp. 126–136.

[131] Junbo Tan, Sorin Olaru, Maria M Seron, and Feng Xu. "Set-based guaranteed active fault diagnosis for LPV systems with unknown bounded uncertainties". In: *Automatica* 128 (2021), p. 109602.

[132] V. Puig. "Fault Diagnosis and Fault Tolerant Control Using Set-Membership Approaches: Application to Real Case Studies". In: *International Journal of Applied Mathematics and Computer Science* 20.4 (2010), pp. 619–635. DOI: 10.2478/v10006-010-0046-y.

[133] J. Blesa and V. Puig and J. Saludes. "Identification for Passive Robust Fault Detection Using Zonotope-Based Set-Membership Approaches". In: *International Journal of Adaptive Control and Signal Processing* 25.9 (2011), pp. 788–812.

[134] Vasso Reppa, Marios M Polycarpou, Christos G Panayiotou, et al. "Sensor fault diagnosis". In: *Foundations and Trends® in Systems and Control* 3.1-2 (2016), pp. 1–248.

[135] P. Valiauga and X. Feng and M. E. Villanueva and R. Paulen and B. Houska. "Set-Membership Estimation Using Ellipsoidal Ensembles". In: *IFAC-PapersOnLine* 54.3 (2021), pp. 596–601.

[136] Shuang Zhang, Vicenç Puig, and Sara Ifqir. "Robust LPV Fault Diagnosis Using the Set-Based Approach for Autonomous Ground Vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* (2024).

[137] Afef Fekih and Darlene Devariste. "A fault-tolerant steering control design for automatic path tracking in autonomous vehicles". In: *2013 American control conference*. IEEE. 2013, pp. 5146–5151.

[138] Tenglong Huang, Huihui Pan, Chi Zhang, and Weichao Sun. "Path Planning and Fault-tolerant Control Based on Resistance Network for Autonomous Driving". In: *2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI)*. IEEE. 2020, pp. 158–162.

[139] Mohamed Ryad Boukhari, Ahmed Chaibet, Moussa Boukhnifer, and Sébastien Glaser. "Sensor fault tolerant control strategy for autonomous vehicle driving". In: *2016 13th International Multi-Conference on Systems, Signals & Devices (SSD)*. IEEE. 2016, pp. 241–248.

[140] Sechan Oh, Hakjoo Kim, Munjung Jang, Jongmin Lee, Kwangseok Oh, and Kyongsu Yi. "Sliding Mode Approach for Partitioned Cost Function-based Fault-Tolerant Control of Automated Driving". In: *2021 21st International Conference on Control, Automation and Systems (ICCAS)*. IEEE. 2021, pp. 1732–1736.

[141] Bong Seok Park. "Neural network-based fault-tolerant control of underactuated surface vessels". In: *Mathematical Problems in Engineering* 2015 (2015).

[142] Xu Jin. "Fault tolerant finite-time leader–follower formation control for autonomous surface vessels with LOS range and angle constraints". In: *Automatica* 68 (2016), pp. 228–236.

[143] Maria Letizia Corradini, Andrea Monteriu, and Giuseppe Orlando. "An actuator failure tolerant control scheme for an underwater remotely operated vehicle". In: *IEEE Transactions on Control Systems Technology* 19.5 (2010), pp. 1036–1046.

[144] Vincent Cocquempot, Roozbeh Izadi-Zamanabadi, Marcel Staroswiecki, and Mogens Blanke. "Residual generation for the ship benchmark using structural approach". In: *UKACC International Conference on Control'98 (Conf. Publ. No. 455)*. IET. 1998, pp. 1480–1485.

[145] Alessandro Freddi, Sauro Longhi, and Andrea Monteriu. "Actuator fault detection system for a remotely operated vehicle". In: *IFAC Proceedings Volumes* 46.33 (2013), pp. 356–361.

[146] Abhishek Dhyani, Rudy R Negenborn, and Vasso Reppa. "A Multiple Sensor Fault Diagnosis Scheme for Autonomous Surface Vessels". In: *12th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS 2024)*. 2024.

[147] Alessandro Baldini, Riccardo Felicetti, Alessandro Freddi, Sauro Longhi, and Andrea Monteriù. "Actuator fault tolerant control via active fault diagnosis for a remotely operated vehicle". In: *IFAC-PapersOnLine* 55.6 (2022), pp. 310–316.

[148] Alessandro Baldini, Riccardo Felicetti, Alessandro Freddi, Sauro Longhi, and Andrea Monteriù. "Fault Tolerant Control for Remotely Operated Vehicles with Thruster Faults using Nonlinear Disturbance Observers". In: *IFAC-PapersOnLine* 55.31 (2022), pp. 275–280.

[149] Bong Seok Park and Sung Jin Yoo. "Fault detection and accommodation of saturated actuators for underactuated surface vessels in the presence of nonlinear uncertainties". In: *Nonlinear Dynamics* 85 (2016), pp. 1067–1077.

[150] Yewen Gu, Julio Cesar Goez, Mario Guajardo, and Stein W Wallace. "Autonomous vessels: state of the art and potential opportunities in logistics". In: *International Transactions in Operational Research* 28.4 (2021), pp. 1706–1739.

[151] Ivan Berman, Enrica Zereik, Aleksandr Kapitonov, Fabio Bonsignorio, Alisher Khassanov, Aziza Oripova, Sergei Lonshakov, and Vitaly Bulatov. "Trustable Environmental Monitoring by Means of Sensors Networks on Swarming Autonomous Marine Vessels and Distributed Ledger Technology". In: *Frontiers in Robotics and AI* 7 (2020).

[152] Aníbal Matos, Eduardo Silva, José Almeida, Alfredo Martins, Hugo Ferreira, Bruno Ferreira, José Alves, André Dias, Stefano Fioravanti, Daniele Bertin, et al. "Unmanned maritime systems for search and rescue". In: *Search and Rescue Robotics-From Theory to Practice* (2017), pp. 77–92.

[153] A. Tsolakis, D. Benders, O. de Groot, R. R. Negenborn, V. Reppa, and L. Ferranti. "COLREGs-aware Trajectory Optimization for Autonomous Surface Vessels". In: *IFAC-PapersOnLine* 55.31 (2022). 14th IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles CAMS 2022, pp. 269–274.

[154] Laura Ferranti, Lorenzo Lyons, Rudy R Negenborn, Tamás Keviczky, and Javier Alonso-Mora. "Distributed nonlinear trajectory optimization for multi-robot motion planning". In: *IEEE Transactions on Control Systems Technology* (2022).

[155] Oscar de Groot, Bruno Brito, Laura Ferranti, Dariu Gavrila, and Javier Alonso-Mora. "Scenario-Based Trajectory Optimization in Uncertain Dynamic Environments". In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 5389–5396.

[156] Oscar de Groot, Laura Ferranti, Dariu Gavrila, and Javier Alonso–Mora. "Globally Guided Trajectory Planning in Dynamic Environments". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 10118–10124.

[157] Dalei Qiao, Guangzhong Liu, Taizhi Lv, Wei Li, and Juan Zhang. "Marine Vision-Based Situational Awareness Using Discriminative Deep Learning: A Survey". In: *Journal of Marine Science and Engineering* 9.4 (2021).

[158] V Garofano, M Hepworth, and R Shahin. "Obstacle Avoidance and Trajectory Optimization for an Autonomous Vessel Utilizing MILP Path Planning, Computer Vision based Perception and Feedback Control". In: *Proceedings of the International Ship Control Systems Symposium*. Vol. 16. 2022, p. 11.

[159] Thor I Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. John Wiley & Sons, 2011.

[160] Lei Du, Osiris A. Valdez Banda, Floris Goerlandt, Yamin Huang, and Pentti Kujala. "A COLREG-compliant ship collision alert system for stand-on vessels". In: *Ocean Engineering* 218.March (2020).

[161] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[162] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex Optimization*. Cambridge university press, 2004.

[163] Alexander Domahidi and Juan Jerez. *FORCES Professional*. Embotech AG. 2023.

[164] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari. "FORCES NLP". In: *International Journal of Control* (2017), pp. 1–17.

[165] Roger Skjetne, Øyvind Smogeli, and Thor I. Fossen. "Modeling, identification, and adaptive maneuvering of CyberShip II: A complete design with experiments". In: *IFAC Proceedings Volumes* 37.10 (2004). IFAC Conference on Computer Applications in Marine Systems - CAMS 2004, Ancona, Italy, 7-9 July 2004, pp. 203–208.

[166] Anastasios Tsolakis, Rudy R Negenborn, Vasso Reppa, and Laura Ferranti. "Model Predictive Trajectory Optimization and Control for Autonomous Surface Vessels Considering Traffic Rules". In: *IEEE Transactions on Intelligent Transportation Systems* (2024).

[167] Zhe Du, Rudy R Negenborn, and Vasso Reppa. "Cooperative multi-agent control for autonomous ship towing under environmental disturbances". In: *IEEE/CAA Journal of Automatica Sinica* 8.8 (2021), pp. 1365–1379.

[168] Mou Chen, Bing Jiang, and Rongxin Cui. "Actuator fault-tolerant control of ocean surface vessels with input saturation". In: *International Journal of Robust and Nonlinear Control* 26.3 (2016), pp. 542–564.

[169] Xianghua Wang. "Active fault tolerant control for unmanned underwater vehicle with actuator fault and guaranteed transient performance". In: *IEEE Transactions on Intelligent Vehicles* 6.3 (2020), pp. 470–479.

[170] Andreas Johansson, Michael Bask, and Torbjörn Norlander. "Dynamic threshold generators for robust fault detection in linear systems with parameter uncertainty". In: *Automatica* 42.7 (2006), pp. 1095–1106.

[171] M. Tanaskovic and L. Fagiano and R. Smith and M. Morari. "Adaptive Receding Horizon Control for Constrained MIMO Systems". In: *Automatica* 50.12 (2014), pp. 3019–3029.

[172] M. Lorenzen and M. Cannon and F. Allgöwer. "Robust MPC with Recursive Model Update". In: *Automatica* 103 (2019), pp. 461–471.

[173] X. Lu and M. Cannon. "Robust Adaptive Tube Model Predictive Control". In: *2019 American Control Conference (ACC)*. IEEE. 2019, pp. 3695–3701.

[174] A. Didier and A. Parsi and J. Coulson and R. S. Smith. "Robust Adaptive Model Predictive Control of Quadrotors". In: *2021 European Control Conference (ECC)*. IEEE. 2021, pp. 657–662.

[175] J. Köhler and P. Kötting and R. Soloperto and F. Allgöwer and M. A. Müller. "A Robust Adaptive Model Predictive Control Framework for Nonlinear Uncertain Systems". In: *International Journal of Robust and Nonlinear Control* 31.18 (2021), pp. 8725–8749.

[176] Luc Jaulin, Michel Kieffer, Olivier Didrit, and Éric Walter. *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics*. London: Springer-Verlag, 2001.

[177] Hervé Brönnimann, Guillaume Melquiond, and Sylvain Pion. "The Boost Interval Arithmetic Library". In: *Real Numbers and Computers*. Lyon, France, 2003, pp. 65–80.

[178] E. Oki. *GLPK (GNU Linear Programming Kit)*. GitLab repository. Accessed: 2024-05-27. 2012. URL: %7Bhttps://api.semanticscholar.org/CorpusID:63578694%7D.

[179] B. Stellato and G. Banjac and P. Goulart and A. Bemporad and S. Boyd. "OSQP: An Operator Splitting Solver for Quadratic Programs". In: *Mathematical Programming Computation* 12.4 (2020), pp. 637–672. DOI: {10.1007/s12532-020-00179-2}.

[180] A. Tsolakis and L. Ferranti and V. Reppa. *Fault Diagnosis in Nonlinear Systems Using Set Membership Estimation*. YouTube video. Accessed: 2024-11-03. 2024. URL: %7Bhttps://www.youtube.com/watch?v=9HzEh0MJ4vE%7D.

[181] Mario E Villanueva, Rien Quirynen, Moritz Diehl, Benoit Chachuat, and Boris Houska. "Robust MPC via min–max differential inequalities". In: *Automatica* 77 (2017), pp. 311–321.

[182] Daniel Limón Marruedo, Teodoro Alamo, and Eduardo F Camacho. "Input-to-state stable MPC for constrained discrete-time nonlinear systems with bounded additive uncertainties". In: *Proceedings of the 41st IEEE Conference on Decision and Control, 2002*. Vol. 4. IEEE. 2002, pp. 4619–4624.

[183] Gilberto Pin, Davide M Raimondo, Lalo Magni, and Thomas Parisini. "Robust model predictive control of nonlinear systems with bounded and state-dependent uncertainties". In: *IEEE Transactions on automatic control* 54.7 (2009), pp. 1681–1687.

[184] Johannes Köhler, Matthias A Müller, and Frank Allgöwer. "A novel constraint tightening approach for nonlinear robust model predictive control". In: *2018 Annual American control conference (ACC)*. IEEE. 2018, pp. 728–734.

[185] Shuyou Yu, Christoph Maier, Hong Chen, and Frank Allgöwer. "Tube MPC scheme based on robust control invariant set with application to Lipschitz nonlinear systems". In: *Systems & Control Letters* 62.2 (2013), pp. 194–200.

[186] Florian Bayer, Mathias Bürger, and Frank Allgöwer. "Discrete-time incremental ISS: A framework for robust NMPC". In: *2013 European control conference (ECC)*. IEEE. 2013, pp. 2068–2073.

[187] Sumeet Singh, Anirudha Majumdar, Jean-Jacques Slotine, and Marco Pavone. "Robust online motion planning via contraction theory and convex optimization". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5883–5890.

[188] Martin Loehning, Marcus Reble, Jan Hasenauer, Shuyou Yu, and Frank Allgoewer. "Model predictive control using reduced order models: Guaranteed stability for constrained linear systems". In: *Journal of Process Control* 24.11 (2014), pp. 1647–1659.

[189] Veronica Adetola and Martin Guay. "Robust adaptive MPC for constrained uncertain nonlinear systems". In: *International Journal of Adaptive Control and Signal Processing* 25.2 (2011), pp. 155–167.

[190] Martin Guay, Veronica Adetola, and Darryl DeHaan. *Robust and adaptive model predictive control of nonlinear systems*. Institution of Engineering and Technology, 2015.

[191] Guilherme AA Gonçalves and Martin Guay. "Robust discrete-time set-based adaptive predictive control for nonlinear systems". In: *Journal of Process Control* 39 (2016), pp. 111–122.

[192] Brett Thomas Lopez. "Adaptive robust model predictive control for nonlinear systems". PhD thesis. Massachusetts Institute of Technology, 2019.

[193] Brett T Lopez, Jean-Jacques E Slotine, and Jonathan P How. "Dynamic tube MPC for nonlinear systems". In: *2019 American Control Conference (ACC)*. IEEE. 2019, pp. 1655–1662.

[194] D Limon, JM Bravo, T Alamo, and EF Camacho. "Robust MPC of constrained nonlinear systems based on interval arithmetic". In: *IEE Proceedings-Control Theory and Applications* 152.3 (2005), pp. 325–332.

[195] Christian Pek and Matthias Althoff. "Computationally efficient fail-safe trajectory planning for self-driving vehicles using convex optimization". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 1447–1454.

[196] Khaled A Mustafa, Daniel Jarne Ornia, Jens Kober, and Javier Alonso-Mora. "RACP: Risk-Aware Contingency Planning with Multi-Modal Predictions". In: *IEEE Transactions on Intelligent Vehicles* (2024).

[197] Lasse Peters, Andrea Bajcsy, Chih-Yuan Chiu, David Fridovich-Keil, Forrest Laine, Laura Ferranti, and Javier Alonso-Mora. "Contingency games for multi-agent interaction". In: *IEEE Robotics and Automation Letters* (2024).

[198] John P Alsterda, Matthew Brown, and J Christian Gerdes. "Contingency model predictive control for automated vehicles". In: *2019 American control conference (ACC)*. IEEE. 2019, pp. 717–722.

[199] Joel A Paulson, Tor Aksel N Heirung, and Ali Mesbah. "Fault-tolerant tube-based robust nonlinear model predictive control". In: *2019 American Control Conference (ACC)*. IEEE. 2019, pp. 1648–1654.

[200] Johannes Köhler, Matthias A Müller, and Frank Allgöwer. "A nonlinear model predictive control framework using reference generic terminal ingredients". In: *IEEE Transactions on Automatic Control* 65.8 (2019), pp. 3576–3583.

[201] Johannes Köhler, Raffaele Soloperto, Matthias A Müller, and Frank Allgöwer. "A computationally efficient robust model predictive control framework for uncertain nonlinear systems–extended version". In: *arXiv preprint arXiv:1910.12081* (2019).

[202] Dennis Benders, Johannes Köhler, Thijs Niesten, Robert Babuška, Javier Alonso-Mora, and Laura Ferranti. *Embedded Hierarchical MPC for Autonomous Navigation*. 2024. arXiv: 2406.11506 [cs.RO]. URL: https://arxiv.org/abs/2406.11506.

[203] Oscar de Groot, Laura Ferranti, Dariu Gavrila, and Javier Alonso–Mora. "Globally guided trajectory planning in dynamic environments". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pp. 10118–10124.

[204] Anoushka Alavilli, Khai Nguyen, Sam Schoedel, Brian Plancher, and Zachary Manchester. "Tinympc: Model-predictive control on resource-constrained microcontrollers". In: *arXiv preprint arXiv:2310.16985* (2023).

[205] Laura Ferranti, Rudy R Negenborn, Tamás Keviczky, and Javier Alonso-Mora. "Coordination of multiple vessels via distributed nonlinear model predictive control". In: *2018 European Control Conference (ECC)*. IEEE. 2018, pp. 2523–2528.

Glossary

AFT Active Fault Tolerance.

AIS Automatic Identification System.

APF Artificial Potential Field.

ARR Analytical Redundancy Relation.

ASV Autonomous Surface Vessel.

AV Autonomous Vehicle.

AVO Acceleration-Velocity Obstacles.

CMPC Contingency Model Predictive Control.

CNN Convolutional Neural Network.

COLREGs Collision Regulations.

CRAMPC Contingency Robust Adaptive Model Predictive Control.

DBN Dynamic Bayesian Networks.

DoF Degrees of Freedom.

DRVO Dynamic Reciprocal Velocity Obstacles.

DWA Dynamic Window Approach.

EM Emergency.

FD Fault Diagnosis.

FDE Fault Detection and Estimation.

FDI Fault Detection and Isolation.

FDIE Fault Detection, Isolation and Estimation.

FDIR Fault Detection Isolation Reconfiguration.

FPS Feasible Parameter Set.

FSM Finite State Machine.

FSMX Fault Signature Matrix.

FT Fault Tolerance.

FTC Fault Tolerant Control.

GNC Guidance Navigation and Control.

GVO Generalized Velocity Obstacles.

GW Give Way.

IDA-PBC Interconnection Damping Assignment Passivity Based Control.

LMI Linear Matrix Inequalities.

LP Linear Program.

LPV Linear Parameter Varying.

LQR Linear Quadratic Regulator.

LSP Least Squares Problem.

LTL Linear Temporal Logic.

MDP Markov Decision Process.

MPC Model Predictive Control.

MPCC Model Predictive Contouring Control.

MPPI Model Predictive Path Integral Control.

NF Navigation Function.

NMPC Nonlinear Model Predictive Control.

OCP Optimal Control Problem.

ORCA Optimal Reciprocal Collision Avoidance.

ORM Obstacle Restriction Method.

OV Obstacle Vessel.

PID Proportional Integral Derivative.

POMDP Partially Observable Markov Decision Process.

PRM Probabilistic Road Maps.

PVO Probabilistic Velocity Obstacles.

QP Quadratic Program.

RAMPC Robust Adaptive Model Predictive Control.

RAS Researchlab Autonomous Shipping.

RHC Receding Horizon Control.

RMPC Robust Model Predictive Control.

ROS Robot Operating System.

ROV Remotely Operated Vehicle.

RPI Robust Positive Invariant.

RRG Randomly Exploring Random Graphs.

RRT Randomly Exploring Random Tree.

RRT* Optimal Randomly Exploring Random Tree.

RVO Reciprocal Velocity Obstacles.

SDP Semi Definite Program.

SMC Sliding Mode Control.

SME Set Membership Estimation.

SO Stand On.

SVD Singular Value Decomposition.

UPS Unfalsified Parameter Set.

VFH Vector Field Histogram.

VO Velocity Obstacles.

Curriculum Vitæ

Anastasios Tsolakis was born in January 1993 in Patras, Greece. He obtained his Diploma (combined B.Sc. and M.Sc.) degree in Mechanical Engineering from Aristotle University of Thessaloniki (AUTH), Thessaloniki, Greece, in 2018. His Diploma thesis entitled "The effect of power distribution among the wheels on vehicle dynamics" was supervised by Prof. Sotirios Natsiavas. Consequently, he obtained an M.Sc. in Systems & Control at DCSC, Delft University of Technology, in 2021. His thesis was entitled "Distributed IDA-PBC for Nonholonomic Mechanical Systems" under the supervision of Prof. Tamás Keviczky.

In December 2020, he started his PhD in a cohesion project between the departments of Cognitive Robotics (CoR) and Maritime and Transport Technology (M&TT), at the Faculty of Mechanical Engineering, Delft University of Technology, Delft, The Netherlands. In his PhD project, he worked on rule-compliant and fault-tolerant trajectory optimization and control algorithms with application to Autonomous Surface Vessels in cooperation with the Researchlab Autonomous Shipping, under the supervision of Dr. Laura Ferranti (CoR), Dr. Vasso Reppa (M&TT) and Prof. Rudy R. Negenborn (M&TT).

His research interests include robot motion planning, model predictive control, and fault-tolerant control.

List of Publications

Referred Journals

1. **A. Tsolakis**, RR Negenborn, V Reppa, L Ferranti, "Model Predictive Trajectory Optimization and Control for Autonomous Surface Vessels Considering Traffic Rules", IEEE Transactions on Intelligent Transportation Systems, Feb 2024.

Referred Conference Proceedings

1. **A. Tsolakis**, T. Keviczky, "Distributed IDA-PBC for a class of nonholonomic mechanical systems", 2021 IFAC - Modelling, Identification and Control of Nonlinear Systems, Tokyo, Japan.

2. **A. Tsolakis**, D. Benders, O De Groot, RR Negenborn, L Ferranti, "COLREGs-aware trajectory optimization for autonomous surface vessels", 2022 IFAC - Conference on Control Applications in Marine Systems, Robotics and Vehicles, Copenhagen, Denmark.

3. **A. Tsolakis**, L Ferranti, V Reppa, "Thruster Fault Diagnosis for an Overactuated Autonomous Surface Vessel", 2024 IFAC - Symposium on Fault Detection, Supervision and Safety for Technical Processes, Robotics and Vehicles, Ferrara, Italy.

4. **A. Tsolakis**, L Ferranti, V Reppa, "Set-Membership Estimation for Fault Diagnosis of Nonlinear Systems," submitted to European Control Conference, Oct. 2024, to be presented in June 2025.

Included in this thesis.
Award finalist.