ISBN 978-82-326-6136-7 (printed ver.)
ISBN 978-82-326-5770-4 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

Doctoral thesis

Sverre Velten Rothmund

Doctoral theses at NTNU, 2023:63

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

NTNU
Norwegian University of Science and Technology

NTNU

Doctoral theses at NTNU, 2023:63

Sverre Velten Rothmund

Risk Awareness and Control of Autonomous Robots

NTNU
Norwegian University of Science and Technology

Sverre Velten Rothmund

Risk Awareness and Control of Autonomous Robots

Thesis for the degree of Philosophiae Doctor

Trondheim, March, 2023

Norwegian University of Science and Technology
Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology
and Electrical Engineering
Department of Engineering Cybernetics

© Sverre Velten Rothmund

ISBN 978-82-326-6136-7 (printed ver.)
ISBN 978-82-326-5770-4 (electronic ver.)
ISSN 1503-8181 (printed ver.)
ISSN 2703-8084 (electronic ver.)

Doctoral theses at NTNU, 2023:63

PRINTED IN
NORWAY
NO - 1598

NORDIC SWAN ECOLABEL
Printed matter
2041 0731

Printed by Skipnes Kommunikasjon AS

Summary

Enabling autonomous systems to safely operate without direct human supervision requires that the systems have the situation awareness which, for a large part, is presently provided by human operators. A crucial part of this situation awareness is that it considers uncertainty and risk, thereby enabling the operator to make risk-based decisions. The goal of this thesis is to develop new methods for giving robotic systems better risk-awareness, thereby contributing towards enabling safer and more efficient autonomous systems.

To contribute towards this goal two different case studies are considered. The first considers an inspection drone operation, where the goal is to increase the drone's understanding of risks related to its own state and the state of the environment. The second considers autonomous collision avoidance at sea, where the goal is to improve the ability of autonomous ships to understand the intentions of meeting traffic. This thesis presents six novel contributions towards this goal, three for each case study. Four of these are based on articles submitted or published in peer-review journals, and two are based on articles submitted or published in peer-reviewed conference proceedings.

The first chapter on the drone case study contributes towards giving the drone a better understanding of uncertainty related to the location and size of obstacles. Here an occupancy grid map is used to model the uncertainty in the environment caused by the drone using a wide-angle radar for obstacle detection. This map is combined with the navigational uncertainty in the drone to evaluate the risk of collision. A scenario-based model predictive control algorithm is used to choose the optimal path offset while considering risk of collision.

The two next chapters consider giving the drone a holistic understanding of risk. These chapters contribute towards enabling the drone to infer its internal health state and the state of the external environment based on indirect measurements, which is then used for automatic decision-making. The first of these chapter consider discrete decision making where a drone tasked with contact-based ultrasound thickness measurements must choose whether it should perform an inspection, request maintenance, or move to a different task. The next chapter considers continuous decisions that have to be made throughout the mission. Here an online supervisory risk controller modifies safety-critical parameters during flight to ensure an acceptable risk level. To ensure that a holistic risk understanding is achieved, a risk analysis is performed to identify hazardous situations. The risk model used by the controller is then made on the basis of this analysis.

The research on the second case study contributes towards enabling autonomous ships to understand the intentions of meeting traffic. The intention model developed in this thesis enables autonomous ships to identify situations prone to cause misunderstanding and to infer the intentions of the other ships based on the observed behavior from AIS and radar. The model is able to understand when ships act in accordance with the maritime traffic rules (COLREGS) and distinguish between different types of incompliant behavior. The model uses statistics on how ships have historically acted for making inferences and predictions. These statistics can be tailored for the current situation, thereby enabling the model to understand that different types of ships act differently in different environments.

The intention model is tested on simulated scenarios, on historical AIS data, and in controlled field experiments. In the field experiments, the intention model is combined with a probabilistic scenario-based model predictive controller for COLREGS-compliant collision avoidance. These experiments demonstrate how the intention model enables the autonomous ship to fulfill its stand-on obligations when other ships seem to act in accordance with the traffic rules and when and how to break from these obligations to ensure safe collision avoidance.

Most of the works in this thesis use different versions of Bayesian belief networks, such as dynamic Bayesian networks and dynamic decision networks. These networks have enabled the inference capabilities needed to infer the state of states that are not directly observable, and the ability to combine information from different factors affecting the risk or behavior in a single model. This thesis demonstrates how these networks can be used to increase risk awareness and artificial intelligence for different case studies and types of inference and modeling problems.

Sammendrag

Å gjøre autonome systemer i stand til å fungere trygt uten direkte menneskelig overvåking krever at systemene får en situasjonsbevissthet som det i dag hovedsakelig er operatøren som står for. En avgjørende del av denne situasjonsbevisstheten er at den tar hensyn til usikkerhet og risiko, noe som gjør operatøren i stand til å ta risikobeviste beslutninger. Målet med denne avhandlingen er å utvikle nye metoder for å gi robotsystemer bedre risikobevissthet, og dermed bidra til å muliggjøre sikrere og mer effektive autonome systemer.

Avhandlingen tar for seg to forskjellige eksempelstudier for å bidra mot dette målet. Den første omhandler en inspeksjonsdrone hvor målet er å øke dronen forståelse av risikoer relatert til den selv og miljøet rundt den. Den andre omhandler autonom kollisjonsunngåesle til sjøs. Her er målet å forbedre autonome skips evne til å forstå intensjonene til den møtende trafikken. Denne avhandlinga presenterer seks forskjellige bidrag mot det overliggende målet. Tre for hvert eksempelstudium. Fire av disse er basert på artikler sendt inn eller publisert i fagfellevurderte tidsskrifter mens to er basert på artikler sendt inn eller publisert i fagfellevurderte konferanser.

Det første kapittelet på drone studiet bidrar mot å gi dronen en bedre forståelse av usikkerhet i plasseringen og utstrekningen til hindringer. Her blir et *occupancy grid map* brukt for å modellere usikkerheten i miljøet som skyldes at dronen bruker en bredviklet radar for å oppdage hindringer. Dette kartet blir kombinert med navigasjonsusikkerheten til dronen for å evaluere risikoen for kollisjon. En scenario-basert modell prediktiv kontroll algoritme brukes for å velge et optimalt avvik fra den planlagte ruta som tar høyde for risikoen for kollisjon.

De neste to kapitlene tar for seg hvordan man kan gi dronen en holistisk forståelse av risiko. Disse arbeidene bidrar til å gjøre dronen i stand til å identifisere dens interne helsetilstand og tilstanden til det eksterne miljøet basert på indirekte målinger. Denne kunnskapen brukes så for automatisk beslutningstaging. Den første av disse kapitlene tar for seg diskret beslutningstaging. Her skal en drone, som brukes til å måle veggtykkelse med en kontaktbasert ultralydsprobe, vurdere om den skal gjennomføre en inspeksjon, be om vedlikehold, eller bytte til et annet inspeksjonspunkt.

Det neste kapittelet tar for seg beslutninger som må tas kontinuerlig gjennom et oppdrag. Her brukes en overvåkende risikokontroller til å endre sikkerhets kristiske parametre i samtid under opperasjonen for å sikre et akseptabelt risikonivå. For å oppnå en holistisk risikomodell så brukes en risiko analyse for å identifiserer faremomenter. Risikomodellen brukt i den overvåkende risikokontrolleren er laget basert på denne analysen.

Forskningen på det andre eksempelstudiet bidrar med å gjøre autonome skip i stand til å forstå intensjonene til møtende trafikk. Intensjonsmodellen som blir utviklet i denne avhandling, gjør det mulig for autonome skip å identifisere situasjoner som kan føre til missforståelser og å forstå intensjonene til andre skip basert på den observerte oppførselen fra AIS eller rader. Modellen forstår når skip handler i samsvar med de maritime trafikkreglene (COLREGS) og er i stand til å skille mellom forskjellige måter reglene kan bli brutt på. Modellen bruker statistikk på hvordan skip pleier å oppføre seg for å forstå intensjonene og forutse fremtidig adferd. Modellen kan utnytte statstikker som er basert på spesifikke skipstyper eller områder for å bedre forstå intensjonene til den møtende trafikken.

Intensjonsmodellen er testet på simulerte scenarier, på historisk AIS data, og på kontrollerte felteksperimenter. I felteksperimentene blir intensjonsmodellen kombinert med en probabilistisk scenariobasert modellprediktiv kontroller for å oppnå kollisjonsunngåelsesoppførsel som er i samsvar med sjøfartsreglene (COLREGS). Disse eksperimentene demonstrerer hvordan intensjonsmodellen gjør det mulig for autonome skipet å oppfylle sine forpliktelser til å holde kurs og fart, og å forstå når den skal bryte disse forpliktelsene for å sikre en trygg kollisjonsunngåelse.

Mesteparten av bidragene presentert i denne avhandlingen bruker forskjellige versjoner av Bayesiske nettverk, slik som dynamiske Bayesiske nettverk og dynamiske beslutnings nettverk. Disse nettverkene muliggjør evnen til å forstå underliggende tilstander basert på indirekte målinger og å kombinere informasjon fra forskjellige faktorer som påvirker risiko eller adferd i en holistisk modell. Denne avhandlingen demonstrerer hvordan disse nettverkene kan brukes for å øke risikobevistheten og den kunstige intelligensen til robotsystemer for forskjellige eksempelstudier og forskjellige klasser av problemer.

Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of Philosophiae Doctor (PhD) at the Norwegian University of Science and Technology (NTNU), Trondheim.

The work presented has been conducted at the Department of Engineering Cybernetics, NTNU, as part of the UNLOCK project. Professor Tor Arne Johansen from the Department of Engineering Cybernetics has been the main supervisor and Professor Ingrid Bouwer Utne from the Department of Marine Technology the co-supervisor. The work was sponsored by the Research Council of Norway through the UNLOCK project, project number 274441, and through the Centre of Excellence funding scheme, project number 223254, AMOS, and from Kongsberg Maritime through the UTC project.

Acknowledgements

I would first like to thank my main supervisor Tor Arne for giving me the freedom and support to pursue my own goals, and for always supporting whatever weird idea I have. I would like to thank my co-supervisor Ingrid for the great effort done on making me understand how and what risk science is, you really opened my eyes to its usefulness. Thank you both for enabling this PhD.

I would like to extend a warm thank-you to my part-time office mate and UNLOCK project mentor, Christoph. I'm thoroughly impressed with your ability to continue answering my never-ending headache-inducing questions and arguments hour after hour, day after day. Without these countless hours of discussions, I would have never managed to get this PhD off the ground. I am especially grateful for your help in getting me and the project going again when no one else could.

I would like to thank my next partner-in-crime, Trym, for our fruitful and very interesting work on intention inference. We are two guys with strong opinions, which at times could cause some heated debates, but I think this combination was crucial for the end-product turning out how it did. Your efforts in getting the experiments up and running, even with uncountable setbacks, were truly impressive.

Furthermore, I would like to thank the rest of the UNLOCK and ORCAS projects.

Jens Einar, Børge, Thomas, and Ruochen, without our discussions I am sure this work would be in a much sorrower state. I would also like to thank the risk group at Tyholt for all the presentations and discussions, giving me a better understanding of the risk sciences.

Next up, I would like to thank ScoutDI for letting me be and feel like a part of the company, both when I was part-time employed, and even more so when I used your equipment and expertise for my PhD work. I would especially like to thank Morten, Kristain, and Kristoffer for your input and guidance on the work, Nicolai for arranging our cooperation, and Harald and Malshan for help with the experimentation, and everyone else for making me feel very welcome.

Furthermore, I would like to thank all the employees at ITK for making it a great working environment. Both when it comes to academic inputs and discussion, but maybe even more so for making it possible to disconnect in a friendly environment. I would especially like to thank my friends who started together with me, Marianna, Pål, Sivert, Andreas, Martin, Joakim, Mathilde, for having been an ever-present part of not only this PhD but also my master journey. I would also like to thank my colleague and life-coach Håkon for watching out for me and helping me with getting going again when I was thoroughly stuck.

I would like to thank my parents, Irene and Peter, for always being interested in and supportive of my PhD work.

Lastly and most importantly I would like to thank Carina for your everpresent support and help in the hardest times. Without you, the sometimes lonely work of doing a PhD, especially through the pandemic, would have been much harder to bear. Thank you for always being there for me.

Contents

Summary i

Sammendrag iii

Preface v

Contents vii

1 Introduction 3
- 1.1 Motivation . . . 3
- 1.2 Previous research . . . 5
- 1.3 Research questions . . . 6
- 1.4 Contributions and outline . . . 7

2 Background theory 13
- 2.1 Bayesian belief networks . . . 13
- 2.2 Risk . . . 20
- 2.3 Decision making under uncertainty . . . 24
- 2.4 The International Regulations for Preventing Collisions at Sea (COLREGs) . . . 33

I Risk-based autonomy for inspection drones 35

3 Risk-based obstacle avoidance 37
- 3.1 Introduction . . . 37
- 3.2 Overview . . . 39
- 3.3 Drone and control model . . . 41
- 3.4 Heading dynamics . . . 45
- 3.5 Covariance propagation . . . 46
- 3.6 Mapping . . . 47
- 3.7 scenario-based model predictive control (SB-MPC) formulation . . . 49
- 3.8 Simulation . . . 53
- 3.9 Discussion and conclusion . . . 54
- 3.10 Future work . . . 57

4 Risk-based decision making **59**
4.1 Introduction . . . 59
4.2 Problem statement . . . 61
4.3 Method . . . 62
4.4 Case study . . . 68
4.5 Results . . . 72
4.6 Discussion . . . 78
4.7 Conclusion . . . 80

5 Supervisory risk control **81**
5.1 Introduction . . . 81
5.2 Method . . . 84
5.3 Industrial drone inspection . . . 89
5.4 Results . . . 104
5.5 Discussion . . . 112
5.6 Conclusion . . . 116
5.7 STPA scenarios . . . 118

II Intention modelling for collision avoidance at sea **121**

6 Development of intention model **123**
6.1 Introduction . . . 123
6.2 Method . . . 125
6.3 Results . . . 144
6.4 Discussion . . . 153
6.5 Conclusion . . . 157

7 Application of intention model in sea trials **159**
7.1 Introduction . . . 159
7.2 System Architecture and Experimental Setup . . . 164
7.3 The probabilistic scenario-based model predictive control (PSB-MPC) collision avoidance (COLAV) Planning Algorithm . . . 167
7.4 Dynamic Obstacle Intention Inference (DOII) . . . 171
7.5 Experimental Results . . . 175
7.6 Discussion . . . 184
7.7 Conclusion . . . 187

8 Validation using historical AIS data **189**
8.1 introduction . . . 189
8.2 AIS data set . . . 190
8.3 Model . . . 191
8.4 Definitions and prior distributions . . . 191
8.5 Result . . . 193
8.6 Discussion . . . 198
8.7 Conclusion . . . 201

III Concluding remarks **203**

9 Conclusion **205**

9.1 Summary and reflections 205

9.2 Contribution to research questions, and future work 207

9.3 Concluding remarks 210

References **213**

Abbreviations

CR-PS	crossing with the other ship on the port side
CR-SS	crossing with the other ship on the starboard side
GW	give-way
HO	head-on
OT-en	overtaken
OT-ing	overtaking
SO	stand-on
AIS	automatic identification system
BBN	Bayesian belief network
BN	Bayesian network
COG	course over ground
COLAV	collision avoidance
COLREGs	the International Regulations for Preventing Collisions at Sea
CPA	closest point of approach
CPT	conditional probability table
CV	constant velocity
DBN	dynamic Bayesian network
DDN	dynamic decision network
DO	dynamic obstacle
DOII	dynamic obstacle intent inference
DOM	dynamic obstacle manager
DP	dynamic positioning
ENC	electronic navigational chart
GNSS	global navigation satellite system
GPU	graphical processing unit
HAZID	hazard identification
IMU	inertial measurement unit
KF	Kalman filter

LOS	line of sight
MPC	model predictive control
NED	north east down
NTNU	Norwegian University of Science and Technology
PHA	preliminary hazard analysis
PhD	Philosophiae Doctor
POMDP	partially observable Markov decision process
PSB-MPC	probabilistic scenario-based model predictive control
ROS	robotic operating system
RTK	real time kinematics
SB-MPC	scenario-based model predictive control
SOG	speed over ground
STPA	system-theoretic process analysis
UCA	unsafe control action
VRS	virtual reference station

Chapter 1

Introduction

"No human is served with doing slave work. It is not interesting, its tiring, and the payment is low [. . .] The work we can say is not human worthy, it should be automated away."

— Jens Glad Balchen, *interview in Aftenposten 1966, translated from Norwegian*

1.1 Motivation

The quote opening this chapter is from Jens Glad Balchen, the founder of the cybernetics department at NTNU, which summarizes one of the main goals of automation, namely freeing people from slave work. By doing so we can free people from doing boring, dangerous, and repetitive tasks, such that they instead can use their time on activities that bring joy and meaning to their lives and the lives of others. Beyond the main objective of improving the lives of people, replacing manual operations with autonomous robots have further advantages for reducing costs [1], improving performance [2], increasing safety [3], and enabling new types of operations [3], [4]. However, today's robotic systems often rely on human operators to monitor them and to manually intervene if necessary [2], [5]–[7]. Changing the role of the operator from controlling to monitoring doesn't only make the work of the operator less enjoyable, but it can also jeopardize the operator's ability to pay sufficient attention needed for them to intervene if necessary. Not actively being engaged in the control of the system can limit the operator's situation awareness of the system they are monitoring [7]–[9]. This can cause new types of accidents as the reduced situation awareness of the operator can prevent them from understanding if something is wrong and what the problem is, and it can prevent the operator from taking over on short notice if the autonomous system enters a state outside of its operational envelope [7]–[9]. Requiring direct human supervision also limits the type of missions that can be automated. This is the case for missions where communication is limited thereby making it impossible to directly monitor the system in real-time, such as for underwater [4] and space [10] operations. Another case is long-term [11] and multi-agent operations [2] that would otherwise be economically infeasible to continuously and directly monitor.

Enabling autonomous systems to solve these challenges requires automating the situation awareness of the operator as well. To explore this topic we first need an understanding of what this situation awareness consists of. A commonly used model for situation awareness is the three-part model presented in [12]. The first part consists of perceiving elements in the environment. The second of using the perceived information to comprehend the current situation. And the last is to project how the situation will develop in the future. Let's consider an example of an inspection drone operated by a skilled human operator to illustrate this model. The operator may perceive a change in the sound coming from the drone, based on experience and training the operator may comprehend that this may mean that one of the motors on the drone is worn. From this, the operator may project that flying with a worn motor can cause overheating thereby causing motor failure during flight. An equally important aspect of projection is to understand how the available decisions affect the future state. In the drone example, this consists of understanding that continuing to fly with a worn motor can cause it to overheat, while landing and maintaining the drone can solve the problem.

The situation awareness will in almost all cases be imperfect. There will always be measurement uncertainties in the information we perceive, our knowledge will be incomplete when we are attempting to comprehend the situation, and our model is imperfect when we are projecting future states. The drone operator may have misheard the change in sound made by the drone, the sound alone is not enough to know if there is actually a problem with a motor, and even if a motor is worn the operator can't know with certainty that it will fail. But even though the situation awareness is uncertain it is still considered rational to abort the mission and inspect the motors in detail. This is due to the perceived increase in probability together with the knowledge that the consequences can be quite high if a motor fails mid-flight. This illustrates that considering risk in situation awareness, that is having a risk awareness, is something humans do naturally.

Giving robotic systems risk awareness and using it for controlling the robot is the topic of this thesis. To explore this topic, two different case studies are considered.

The first case study, presented in Part I of the thesis, considers an industrial inspection drone. This case study was chosen as these drones are relatively simple making it possible to consider the system as a whole thereby gaining a holistic overview. These systems are also a safe place to start exploring higher levels of autonomy as they often can operate separately from, or far away from, humans making the danger of hurting humans minimal. This section first explores how to consider the uncertainty in the drone's motion and the position of obstacles when avoiding collision. Thereafter two different chapters on giving the drone a holistic risk awareness are presented. The first considers discrete decision making where the system has to decide on what to do next after a task execution attempt fails. The second considers decisions that have to be continuously made throughout the mission such as choosing adequate safety distance to obstacles and choosing adequate speed and acceleration limits.

The second case study, presented in Part II of the thesis, considers autonomous ships. In contrast to part one, this part focuses on a single source of uncertainty in the systems situation awareness, namely how other agents plan to maneuver in a collision encounter. An agent here refers to any entity that interprets and acts upon the same environment as the robotic system, such as other ships. Even though there are rules for how to act at sea [13], it is insufficient to blindly assume that all other ships have interpreted the situation in the same way as us and that they know or plan to follow the rules. Therefore one must pay attention to how the other ship acts (perception), understand if they intend to follow the rules and if they have interpreted the situation in a different manner (comprehension), and use this information to understand how the ship may move in the future (projection). Having a probabilistic understanding of how the other ships will act is a crucial part of considering risk in the situation awareness of autonomous ships.

1.2 Previous research

This section presents a short overview of previous research on using risk or uncertainty in the situation awareness of robotic systems. Each of the following chapters will go further into detail on previous research related to that particular chapter. The goal of this chapter is to give the necessary background for formulating the research questions.

Many publications take risk or uncertainty into account in the situation awareness used for control in one way or another. Some examples are: [14]–[17] which consider a heuristic metric of risk based on a few measurable factors, such as the time until and distance at the closest point of approach between the robotic system and other agents, and/or relative speed between them, [18], [19] which consider the uncertainty in the predicted future trajectories of the robotic system, [20]–[23] which considers uncertainty in the future trajectory of the robotic system and other agents, and [24] which considers uncertainty in the future trajectories of the robotic system and uncertainty in the location of static obstacles.

However, few publications have considered the uncertainty stemming from the intention of other agents. Nevertheless, some examples can be found for ships [25], airplanes [26], cars [27], and pedestrians [28], [29].

Replacing the situation awareness of a human requires a holistic overview of relevant factors affecting the risk. To achieve this, the risk metrics used in the previously presented literature must be combined with all other relevant factors. One way of aggregating risk when doing path planning is using a risk map [30]–[34]. These articles evaluate a separate risk map for each risk influencing factor. The different risk maps can then be combined by for example taking a weighted sum. Another approach, which can model the relationship between different risk influencing factors, is to use a Bayesian belief network (BBN) [35]–[43]. Of these publications only [36]–[40] uses the BBN for on-line decision making. An alternative to BBNs is presented in [44], [45] where a risk graph is used to combine different risk-related properties. A process for ensuring that the risk awareness includes all relevant risk influencing factors is to perform a risk analysis as proposed in [35].

1.3 Research questions

The overarching research question motivating this PhD work is as follows:

> **How can the risk awareness of robotic systems operating in the real world be improved such that they can safely operate without direct human supervision?**

The goal of this PhD work has been to develop new research that contributes towards this overarching question. This thesis will therefore focus on presenting novel research rather than giving a comprehensive review of the existing literature related to this question. To guide us through the research developed in this PhD work, 6 research questions are proposed. These questions are based on identified holes in the literature related to the overarching research questions. The rest of this section will outline the holes in the literature and present the proposed research questions. Thereafter Section 1.4 will outline the thesis and summarize how each chapter contributes towards the 6 proposed research questions.

Much of the surveyed literature considers uncertainty in the position and velocity of other agents. Very few consider uncertainty in the location of static obstacles. Other agents can often be modeled as point masses with a corresponding uncertainty in position and speed. This approach cannot be transferred to static obstacles such as walls, as their shape, location, and size are the relevant sources of uncertainty. The work presented in [24] considers both mapping and position uncertainty, but they do not incorporate the uncertainty stemming from the sensor when building their map. Based on this, the following research question is proposed:

RQ 1 How can uncertainty in static obstacles be modeled when considering the uncertainty in the sensor mapping the obstacle, and how can this be combined with uncertainty in the navigation of the robotic system itself in a risk-based collision avoidance algorithm?

Another single source of uncertainty that has received limited attention is the intention of the other agents. The only other article that was found to consider probabilistic intentions at sea [25] only considers whether the ship is acting in accordance with the own-ship interpretation of the rules. It does not model how disagreements can arise nor does it differentiate between different ways the ship can act in incompliant ways. Based on this, the following research questions are proposed:

RQ 2 How can the intention of other agents at sea be modeled and inferred in greater detail so that the future behavior of the agent can be predicted with higher accuracy?

RQ 3 How can an improved situation awareness of the other ships' intentions improve the collision avoidance capabilities of autonomous ships?

Making a probabilistic model that combines multiple factors to give a holistic risk awareness for operational decision-making has had limited attention in the literature. Among the surveyed literature, only [38], [40] consider this topic for robotic systems. [38] considers choosing a landing location for an unmanned aircraft system by combining many different factors affecting the suitability of the location in a BBN. [40] considers using the dynamic counterpart of BBNs, dynamic Bayesian network (DBN), for fault identification, prognosis, and recovery. None of these work focus on risk awareness for operational decision-making under nominal conditions. This opens up the following research questions:

RQ 4 How can a robotic system build and use risk awareness for operational decision-making for considering which task or action to do next?

RQ 5 How can a robotic system build and use risk awareness for continuous decisions that have to be made during a task execution?

To increase the likelihood that all relevant factors are included in the situation awareness, a risk analysis can be performed as proposed in [35]. However, [35] does not consider how the resulting risk model can be used for control. This opens up the following research question:

RQ 6 How can a situation awareness model based on the results from a risk analysis be used to ensure safer autonomy?

Note that during the development of this PhD my colleagues in the UNLOCK and ORCAS projects have explored RQ 5 and RQ 6 as well for the underwater and autonomous ship domains. At the time of writing, this has resulted in the publications [36], [37]. How the different chapters distinguish themselves from these publications is discussed in the corresponding chapters.

1.4 Contributions and outline

This thesis consists of 9 chapters where Chapters 3 to 8 each present the result of one publication developed during this PhD. Each chapter starts with a statement stating the contribution of each author.

The thesis consists of two parts. Part I, consisting of Chapters 3 to 5, contains research on enabling safer autonomy for an industrial inspection drone. Part II, consisting of Chapters 6 to 8, contains research on an intention model used to for collision avoidance at sea.

Background material relevant to multiple chapters is given in Chapter 2 and concluding remarks are given in Chapter 9.

The rest of this chapter outlines the research and contribution of the chapters presenting novel results towards the research questions presented in Section 1.3.

Chapter 3 - Risk-based obstacle avoidance

[46] **S. V. Rothmund** and T. A. Johansen, "Risk-Based Obstacle Avoidance in Unknown Environments using Scenario-Based Predictive Control for an Inspection Drone Equipped with Range Finding Sensors," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, Jun. 2019, pp. 221–230. DOI: 10.1109/ICUAS.2019.8797803

This chapter develops an obstacle avoidance strategy for an inspection drone equipped with wide-angle range-finding sensors, such as radar or sonar. The obstacle avoidance strategy uses scenario-based model predictive control (SB-MPC) [14] to find an optimal path offset that ensures an acceptable probability of collision. The probability of collision is evaluated considering both the position uncertainty of the drone and the mapping uncertainty stemming from the measurement uncertainty in mapping sensor. The mapping uncertainty is modeled with an occupancy grid map. 2D simulations show that the drone is able to mitigate risk by changing speed and taking detours to avoid flying in potentially dangerous areas.

This chapter contributes towards RQ 1 by developing a risk-based framework for local obstacle avoidance in unknown environments that incorporates both uncertainties in the environment due to sensor uncertainty and uncertainty in the vehicle's position due to navigation uncertainty. The algorithm developed in [14] for collision avoidance between ships is adapted to the drone case and modified to use uncertainties in the drone state and the environment to calculate the probability of collision.

Chapter 4 - Risk-based decision making

[47] **S. V. Rothmund**, C. A. Thieme, I. B. Utne, and T. A. Johansen, "A Bayesian Approach to Risk-Based Autonomy for a Robotic System Executing a Sequence of Independent Tasks," *Submitted to Journal of Intelligent & Robotic Systems*, 2022. DOI: 10.36227/techrxiv.14054141.v3

This chapter develops a risk-based decision-making system for a robotic system executing a series of independent tasks. A dynamic decision network (DDN) [48] is used to decide if and how a task should be executed, and whether the system should be maintained. If a task execution is attempted then information on how the execution went is given to the DDN. By considering the results of different execution attempts together with the choices made by the system the DDN is able to identify the state of underlying causal factors that can prevent a safe and successful task execution. Information on the causal factors is used to evaluate the risk and gain of executing different actions. A simulation case study of an industrial inspection drone performing contact-based inspection is used to demonstrate the capabilities of the resulting system.

The risk-based decision-making system developed in this chapter contributes towards RQ 4 by increasing the robotic system's risk awareness beyond what is directly observable and by enabling it to reason about risk when making decisions. The case study demonstrates that the resulting system is able to infer the presence of faults with the drone and adverse environmental conditions. This information enables the drone to act cautiously in potentially dangerous situations, request maintenance when needed, and identify and handle past mistakes.

Chapter 5 - Supervisory risk control

[49] **S. V. Rothmund**, C. A. Thieme, I. B. Utne, and T. A. Johansen, "Supervisory Risk Control with Application to Industrial Drone Inspection," *Submitted to Autonomous Robots*, 2022. DOI: 10.36227/techrxiv.21287334

This chapter develops and experimentally tests a supervisory risk controller used to increase the safety of drone operations. Its task is to monitor the state of the drone and environment and to use this information to automatically change safety-critical parameters in real-time during operation. A case study of a tethered industrial inspection drone is considered. A system-theoretic process analysis (STPA) [50] is performed to identify how the system can fail. A DDN, used as an online risk model, is built based on the results of the STPA. An optimization approach is used to choose an optimal parameter configuration that ensures that the risk level evaluated with the DDN is acceptable. A mission abort recommendation is sent to the human operator if no parameter configuration can ensure an acceptable risk. The supervisory risk controller's situation awareness is continuously forwarded to the human operator to increase their ability to understand and handle potentially dangerous situations.

This chapter contributes towards RQ 5 and RQ 6 by developing a supervisory risk controller based on a risk analysis that continuously monitors a drone operation and modifies safety-critical parameters. The chapter further develops the method proposed in [35] by explicitly modeling measurements, by modeling how states develop over time, and by modeling the effect of decisions. This development enables the system to identify the state of causal factors that cannot be directly observed by combining information from different measurements over time. The chapter experimentally validates the supervisory risk controller on an inspection drone case study.

Chapter 6 - Development of intention model

[51] **S. V. Rothmund**, T. Tengesdal, E. F. Brekke, and T. A. Johansen, "Intention modeling and inference for autonomous collision avoidance at sea," *Ocean Engineering*, vol. 266, p. 113 080, Dec. 2022. DOI: 10.1016/j.oceaneng.2022.113080

This chapter develops a probabilistic model of the intentions of other ships in collision encounters. The underlying intentions that define the other ships' behavior are inferred by combining real-time measurements in a DBN. The evaluated intentions can either be directly used in a collision avoidance algorithm or the DBN can be used to evaluate the probability of meeting traffic following different candidate trajectories. The model considers multiple different intention states to describe the different ways a ship can interpret and conflict with the behavioral rules outlined in the International Regulations for Preventing Collisions at Sea (COLREGs)[13]. The prior probability distributions of the intention states can be adapted to the current situation based on observable characteristics such as location and relative ship size. The resulting model is able to identify situations that are prone to cause misunderstandings and infer the state of multiple intention variables that describe how the ship is likely to behave. Different collision avoidance algorithms can use the resulting intention information to better know if, when, and how to act.

This chapter contributes towards RQ 2 by developing a modeling framework that considers how underlying causes affect a ship's maneuvering behavior. The framework can infer the state of multiple different intention variables based on measured properties. This enables the model to describe the future maneuvering behavior of other ships with higher fidelity than simply being COLREGs compliant or not.

Chapter 7 - Application of intention model in sea trials

[52] T. Tengesdal, **S. V. Rothmund**, E. A. Basso, T. A. Johansen, and H. Schmidt-Didlaukies, "Obstacle Intention Awareness in Automatic Collision Avoidance: Full Scale Experiments in Confined Waters," *Submitted to Field Robotics*, 2022

In this chapter, the intention model introduced in Chapter 6 is combined with the probabilistic scenario-based model predictive control (PSB-MPC) developed in [20], [53] to create an intention aware collision avoidance algorithm. The resulting system is experimentally tested in the Trondheim harbor with the Milliampere ferry. The experiments emphasize on hazardous situations where intention information is both useful and necessary to avoid high collision risk.

This chapter contributes towards RQ 3 by incorporating the intention model in a collision avoidance system thereby increasing its situation awareness. Furthermore, the chapter experimentally validates the model through the sea trials and demonstrates how the resulting collision avoidance system adheres to COLREGs rules 7-8 and 13-17 in a diverse set of situations.

Chapter 8 - Validation using historical AIS data

[54] **S. V. Rothmund**, H. E. Haugen, G. D. Veglo, E. F. Brekke, and T. A. Johansen, “Validation of ship intention model for maritime collision avoidance control using historical AIS data,” *Submitted to ECC*, 2023

In this chapter, the intention model introduced in Chapter 6 is validated on historical data from ships along the Norwegian coast. A series of COLREGs relevant encounters from historical data available from the automatic identification system (AIS) [55] is considered. Some of the example distributions used in Chapter 6 are replaced with empirical distributions extracted from the AIS data. The intention model applied to all ships in the encounter, feeding it measurements as if the encounters were happening in real time. How the belief regarding the ship’s underlying intentions develop throughout the encounters are analyzed.

This chapter contributes towards RQ 2 by validating that the intention model works with empirical distributions and works on historical ship encounters. This chapter demonstrates the capabilities of the model and highlights the weaknesses that should be studied further.

Chapter 2

Background theory

2.1 Bayesian belief networks

This section will give an introduction to BBNs, sometimes also called Bayesian networks (BNs), which are heavily used throughout this thesis. First, a motivating example of the type of inference capabilities Bayesian networks possess is given in Section 2.1.1. Thereafter, in Section 2.1.2, the underlying equations enabling this inference capability are given. Lastly, Section 2.1.3 outlines some common extentions to BBNs.

2.1.1 Overview with motivating example

Let's start with the simple example shown in Figure 2.1 which attempts to model whether a coworker is late for work. The figure consists of nodes (circles), which represent the states we are interested in, and arcs (pointed arrows), which describe dependencies. One of the nodes representing whether the coworker is late and the two other possible reasons for the coworker being late. In this example, these are the coworker oversleeping and the bus being late. The directed arcs, or arrows, in the graph indicate that both oversleeping and the bus being late influence whether the coworker is late for work.

The topology (layout of nodes and arrows) tells us how different nodes depend on each other, but to be able to calculate the probability that the coworker is late we need to supply numbers to the model. The numbers we need are the prior belief that the coworker will oversleep, here chosen to be 5%, and the prior belief that the bus is late, here chosen to be 20%. These types of probabilities are called priors and represent our beliefs before any information is gathered. The second type of information we need to provide is how the nodes relate to each other, in this case, how begin late depends on oversleeping and the bus being late. Here we apply a simple model where there is a 100% chance of being late if the coworker either oversleeps or the bus is late, and a 5% chance of the coworker being late even when she does not oversleep and all the busses are on time. This information

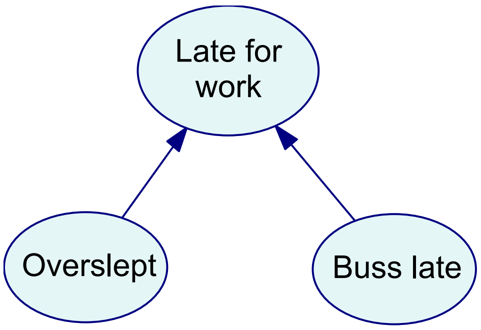
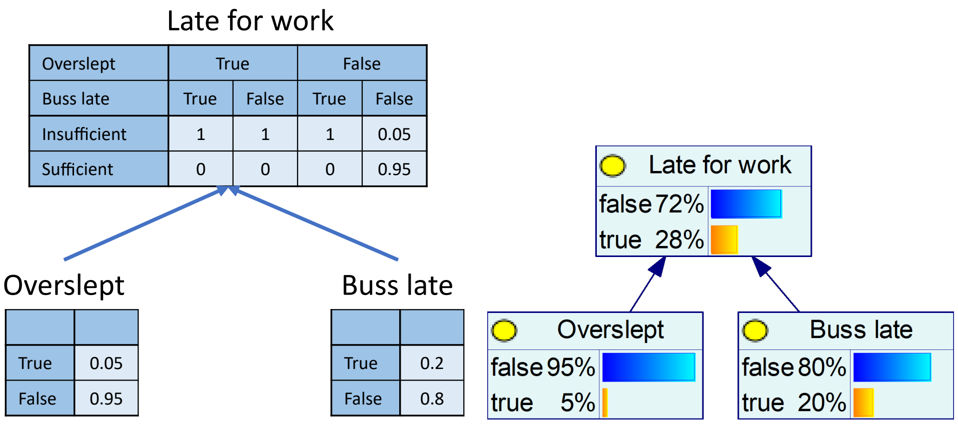


Figure 2.1: Topology of a BBN modelling a coworker being late for work.



Late for work

Overslept	True		False	
Buss late	True	False	True	False
Insufficient	1	1	1	0.05
Sufficient	0	0	0	0.95

Overslept

True	0.05
False	0.95

Buss late

True	0.2
False	0.8

(a) A representation of the late for work BBN showing all CPTs.

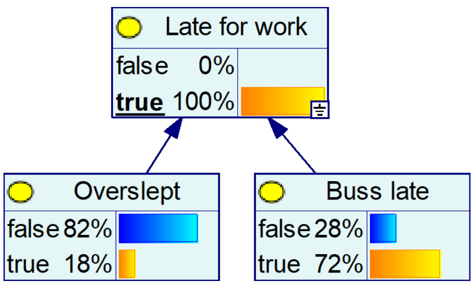
(b) The probability of being “late for work” evaluated with the BBN.

Figure 2.2

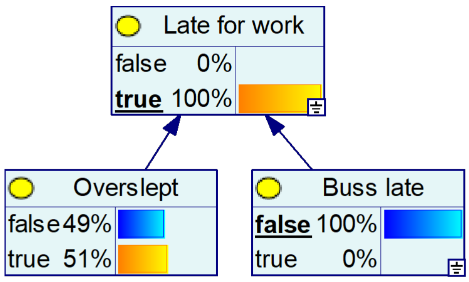
is summarized in the tables shown in Figure 2.2(a), which are called conditional probability tables (CPTs).

The first thing we can use a bayesian network for is to evaluate, given our prior probabilities, what the likelihood is that the coworker will be late. The result of this evaluation is shown in Figure 2.2(b).

The next thing we can do is to use the BBN to understand the causes of an observation. If we on a particular day observe that the coworker is late, then we can give this information to the Bayesian network as evidence. Evidence is the information that a particular node is in a particular state, in this case, that "late for work" is "true". Now if with this new information, we can update our beliefs regarding whether the coworker did oversleep today and whether the bus was late today. From Figure 2.3(a) we can see that given our proior beliefs, we think it’s much more likely that the coworker being late was caused by the buss being late rather than the coworker oversleeping.



(a) BBN evaluated after inserting evidence that "late for work" is in the state "true".



(b) BBN evaluated after inserting evidence that "late for work" is in the state "true" and "buss late" in the state "false".

Figure 2.3

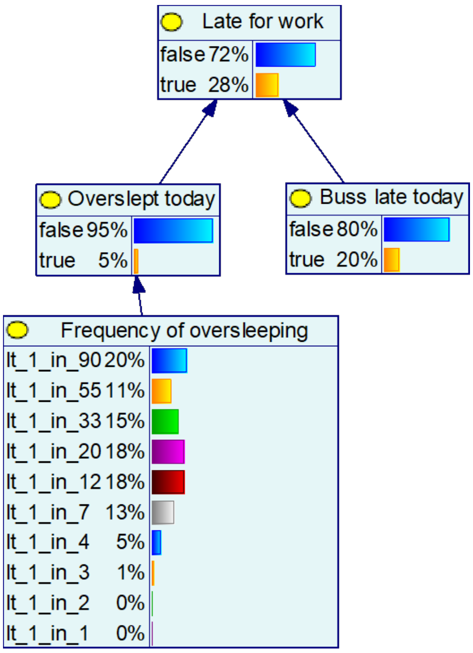
Furthermore, we can use BBNs to combine multiple pieces of evidence. Let's say that we meet another college that took the same bus and could testify that the bus was on time today. If we insert this information into the BBN and update our beliefs we get the evaluation given in Figure 2.3(b). From this figure, we can see that it's more likely that the coworker has overslept when we also know that the bus is not late. We are still not 100% sure that the coworker has overslept as there was a chance of the coworker being late even if she did not oversleep and the bus was on time. This probability can be interpreted as other reasons for the coworker to be late, not considered by our model.

Even though we now are 51% certain that the coworker is late today, this does not mean that we expect the coworker to be late around every second day. This would be unreasonable to propose after only one observation. Instead, this probability represents the strength of our belief that she has overslept today. This way of thinking about probability is often called the Bayesian interpretation of probability [56].

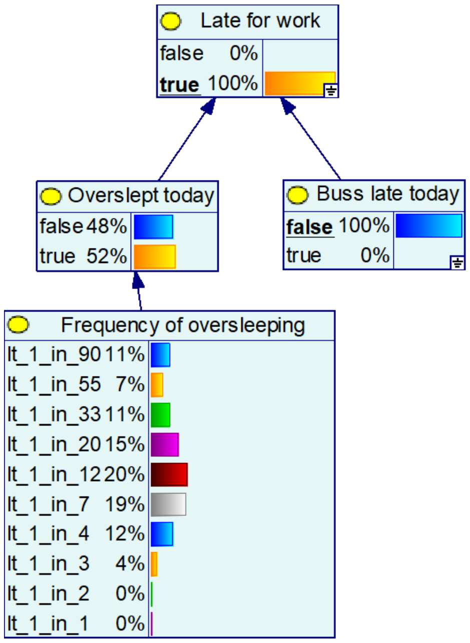
Now if we are interested in estimating the underlying frequency of our college over-sleeping we can redefine our network as shown in Figure 2.4(a). We have here split the oversleeping node into two, one representing the average frequency of the coworker oversleeping and the other whether the coworker oversleeps on this particular day. Now instead of specifying our belief that the coworker will oversleep today, we instead insert our belief regarding the different frequencies of the coworker oversleeping. This belief is represented by the probability distribution shown on the "frequency of oversleeping" node in Figure 2.4(a).

With the new network, if we observe on a particular day that the coworker is late, as seen in Figure 2.4(b), the probability that she has overslept today changes a lot, while our belief regarding the frequency of her oversleeping does not change as much, as we would expect based on only one day of observation.

Using only a single observation is not sufficient to get a high belief in our coworker



(a) A BBN considering the underlying frequency of oversleeping. "lt_1_in_90" reads as "frequency of occurance i less than 1 in 90".



(b) A BBN considering the underlying frequency of oversleeping where evidence is inserted (marked in bold).

oversleeping with any particular frequency. To consider multiple days we need to expand the BBN to a DBN. A DBN works in exactly the same manner as a BBN, but we have repeated the network for each day we are gathering data, as shown in Figure 2.5. In addition, we need to specify the transition probabilities, that is how our belief regarding the coworker's oversleeping frequency should develop over time. In this example, the transition probability may represent that the coworker can change their behavior, by for example improving their day-night cycle, which will make oversleeping less likely.

We will return to the concept of having probability distribtuion over frequencies in Chapter 4, where we want to infer with what frequency different underlying faults can cause a task to fail, and in Chapter 5, where we want to model with what frequency different scenarios will occur.

2.1.2 Underlying calculations

To presenting the underlying calculations, lets instead consider the more abstract case shown in Figure 2.6, where the CPTs are formulated as in Figure 2.7.

All of the different uses of BBNs presented in the previous section are the exact same mathematical problem. That is, what is the probability that a node, A, is in a state a, when we have evidence $\mathcal{E}$.

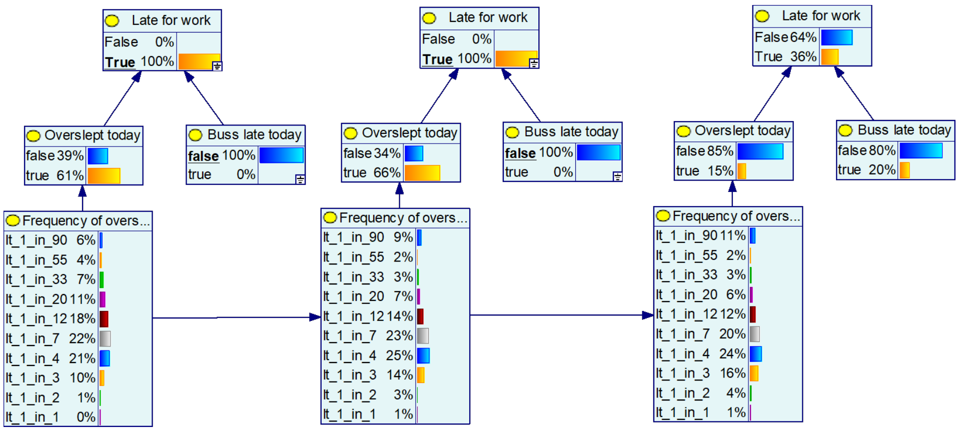


Figure 2.5: Example of a DBN considering multiple days of observations.

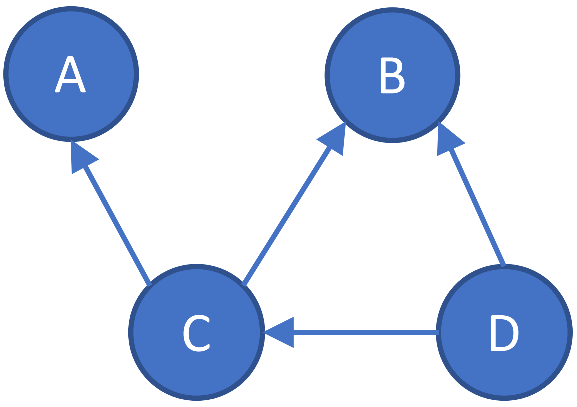
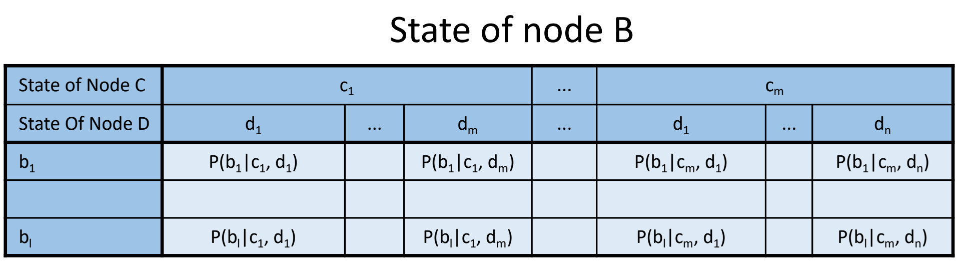


Figure 2.6: Example of a BBN consisting of 4 nodes.



State of node B

State of Node C	c_1		...	c_m		
State Of Node D	d_1	...	d_m	...	d_1	...	d_n
b_1	$P(b_1 \mid c_1, d_1)$		$P(b_1 \mid c_1, d_m)$		$P(b_1 \mid c_m, d_1)$		$P(b_1 \mid c_m, d_n)$
b_l	$P(b_l \mid c_1, d_1)$		$P(b_l \mid c_1, d_m)$		$P(b_l \mid c_m, d_1)$		$P(b_l \mid c_m, d_n)$

Figure 2.7: Example of a CPT for node B in figure 2.6 when node B take on the discrete set of state $\{b_1...b_l\}$, C the states $\{c_1...c_m\}$, and D the states $\{d_1...d_n\}$.

$$P(A = a|\mathcal{E}) \tag{2.1}$$

As previously stated, evidence is the knowledge that a particular node is in a particular state. If our evidence for example consists of B being in state b, and D in state d, then we can write:

$$P(A = a|\mathcal{E}) = P(A = a|B = b, D = d) \tag{2.2}$$

Using the chain rule in reverse, this can be rewritten as:

$$P(A = a|\mathcal{E}) = \frac{P(A = a, B = b, D = d)}{P(B = b, D = d)} \tag{2.3}$$

As the denominator is a constant value it can be replaced with a scalar η. This scalar can be found after calculating the probability that A is in any of it states $a_1, a_2, ..., a_j$, by ensuring that $\sum_{a_i} P(A = a_i|\mathcal{E}) = 1$. This results in the following equation:

$$P(A = a|\mathcal{E}) = \eta P(A = a, B = b, D = d) \tag{2.4}$$

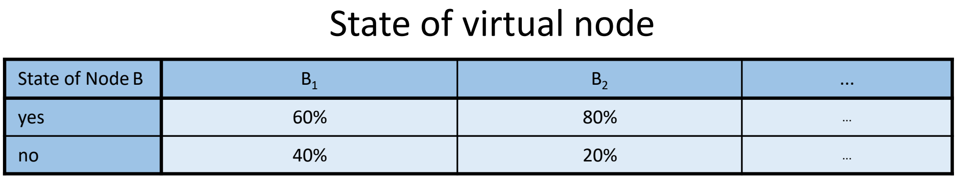
The next step is to include all nodes that are so for not considered in the equation. For this example, it is node C. This can be achieved using the law of total probability.

$$P(A = a|\mathcal{E}) = \eta \sum_{c_i} P(A = a, B = b, C = c_i, D = d) \tag{2.5}$$

The term $P(A = a, B = b, C = c_i, D = d)$ is the probability that the network is in a particular state. This term can be calculated using the chain rule while adhering to the dependencies specified by the arcs in the BBN:

$$\begin{aligned} P(A = a, B = b, C = c_i, D = d) =& P(A = a|B = b, C = c_i, D = d) \\ & P(B = b|C = c_i, D = d) \\ & P(C = c_i|D = d)P(D = d) \qquad (2.6) \\ =& P(A = a|C = c_i)P(B = b|C = c_i, D = d) \\ & P(C = c_i|D = d)P(D = d) \qquad (2.7) \end{aligned}$$

The last step utilizes the fact that A is conditionally independent on both B and D given C, which can be seen from the arcs in Figure 2.6. All the resulting terms, $P(A = a|C = c_i)$, $P(B = b|C = c_i, D = d)$, $P(C = c_i|D = d)$, and $P(D = d)$ are given by the CPTs.


State of virtual node

State of Node B	B_1	B_2	...
yes	60%	80%	...
no	40%	20%	...

Figure 2.8: Example of a CPT for a virtual node giving uncertain evidence on node B.

Performing exact inference in BBNs is in the worst case an NP-hard problem [57], but different established solvers exist for quickly finding the exact solution in medium-sized networks [58] as well as solvers for finding approximate solutions [59]. Different software tools such as [60], [61] have implemented different general solvers. As these solvers do all of the necessary computations the rest of the thesis will not focus on how the inference is done.

2.1.3 Extentions

BBN can also handle uncertain evidence, called virtual evidence. Instead of stating that, for example, node B is in state b_1 with 100% certainty, virtual evidence will instead state that a measurement was made which has, for example, 60% chance of occurring if node B is in state b_1 and 80% if node B is in state b_2. Virtual evidence can be implemented by temporarily inserting a new node with the CPT given in Figure 2.8. Normal evidence can then be inserted on the virtual node stating that it is in the "yes" state. A thorough explanation of virtual evidence can be found in [62]. The use of virtual evidence is supported in [60].

As previously stated, BBNs can be made into DBN by repeating the network for each time step and connecting the nodes based on how they depend on each other across time. There is no difference in evaluating a BBN and a DBN.

Another extension that can be made is to consider decisions as well as dynamics. This would create a DDN. This is achieved by letting some of the nodes represent decision variables and including the decision in the list of evidence. As there is no difference between a node representing a decision and all other nodes, DDNs can be evaluated in the same manner as BBNs and DBNs.

More information on BBNs and DDNs can be found in [48], [63].

2.2 Risk

As this thesis considers the topic of risk awareness a short introduction to topics related to risk is needed. To start this discussion, let's consider the concept of a hazard or hazardous event. A hazard is a source of risk [64], which can more precisely be defined as "a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to a loss" [50]. Loss is here defined as losing something of value to a stakeholder [50].

When analyzing a hazard the concept of the bowtie diagram, shown in Figure 2.9, can be useful. This diagram demonstrates that there are multiple causal factors that can cause a hazardous event to occur and that a hazardous event can cause a myriad of different losses. In analysis, the focus is often placed on the hazards rather than the loss directly. This is due to the hazard being a part of the system that we can control, while the occurrence of a loss will depend on environmental conditions that are outside our control [50]. The term consequence will be used as a collective term for the different types of losses and how severe the losses are.

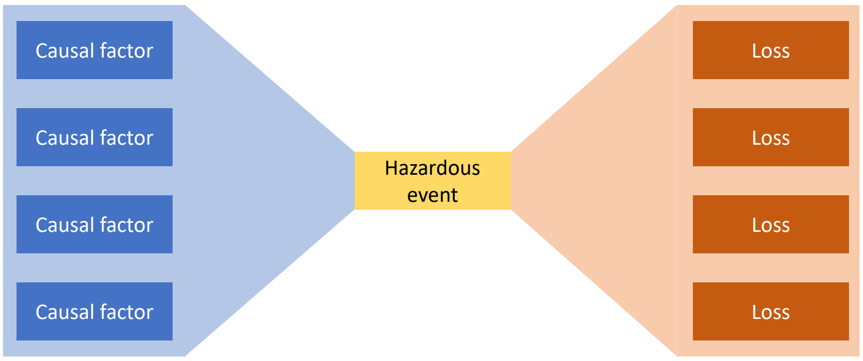


Figure 2.9: Bowtie diagram that shows that there are multiple causal factors related to a hazardous event and multiple possible losses caused by the event.

Analyzing risk, compared to a hazard analysis, requires some manner of quantification. According to [64], risk is the combined answer to the following questions: What can go wrong? How likely is it? And what are the consequences? More precisely they give a definition of risk as

$$R = \{(s_i, P_i, c_i)\} \tag{2.8}$$

That is, risk is the set of scenarios s_i, the probability of the scenario occurring, P_i, and the corresponding consequences, c_i, where the index i enumerates all scenarios. Furthermore, they expand this definition of risk to also include our uncertainty. Instead of giving a probability of a scenario occurring they instead consider our belief of the scenario occurring with different frequencies. In the same manner, as for the coworker being late example discussed in section section 2.1, a probability distribution is given over the frequencies of the scenario occurring. This leads to the following definition of risk where ϕ represents frequency:

$$R = \{(s_i, p_i(\phi), c_i)\} \tag{2.9}$$

Note that the probability term used in Equation (2.8) can be found by taking the expected value of the frequency formulation, that is:

$$P_i = \int \phi \, p_i(\phi) \, d\phi \tag{2.10}$$

This equation reveals that it's more correct to think of the term P_i in Equation (2.8) as the expected frequency rather than a probability. For discrete events such as whether or not a rocket will explode on take-off, both interpretations of P_i work. We can talk about the expected frequency of rockets exploding on take-off (given that we regularly send up rockets), and talk about the probability of this particular rocket exploding on take-off. But for phenomenas that cannot be seen as discrete events the probability formulation has some problems. If we are for example considering the collision risk of a ship, the question "what is the probability of collision" does not make sense unless we specify the time span we are considering. This can be achieved by for example changing the question to the probability of colliding within 10 years or considering a discrete event such as one voyage. With the frequency formulation, we naturally talk about the expected frequency of collision being for example once per 100 years.

In this definition, the risk is not a single number. It is simultaneously a description of what can go wrong, together with a quantification of losses, frequencies, and uncertainties. In [64] it is cautioned against reducing risk to a single number as this process leads to loss of information, such as a very likely scenario with very low consequences receiving the same risk value as a very unlikely but high consequence scenario. But to use risk in automatic decision making it must be possible to compare the risk of different actions. This will, in most applications, require mapping the risk to a single value, even though this will necessarily lead to a loss of information. One way of doing this mapping is to evaluate the expected risk

$$\begin{aligned} E[R] &= \sum_i \int \phi L(c_i) p_i(\phi) d\phi && (2.11) \\ &= \sum_i L(c_i) P_i && (2.12) \end{aligned}$$

That is to sum over the expected frequency multiplied with the consequences for all scenarios, i, where $L(c_i)$ quantifies consequence as a single number.

Note that in this formulation a scenario is a specific chain of events that causes exactly one outcome. This means that there are infinite scenarios with marginal differences. If we instead want to consider broader scenario categories we need to consider the probability of observing different consequences. The most general formulation given in [64] considers the probability of observing a consequence, c, with a particular frequency:

$$R = \{(s_i, p_i(\phi, c))\} \tag{2.13}$$

The expected risk is then

$$E[R] = \sum_i \int L(c) \int \phi \, p_i(\phi, c) \, d\phi \, dc \tag{2.14}$$

$$\tag{2.15}$$

A less general formulation given in [64] considers the probability of observing the scenario with different frequencies, $p_i(\phi)$, and the probability of observing different consequences if the scenario occurs, $\zeta_i(c)$.

$$R = \{(s_i, p_i(\phi), \zeta_i(c)\} \tag{2.16}$$

In this case, the expected risk can be formulated as follows where $E_i[L(c)]$ is the expected consequence of scenario i.

$$E[R] = \sum_i \int \phi \, p_i(\phi) \, d\phi \int L(c) \, \zeta_i(c) \, dc \tag{2.17}$$

$$= \sum_i P_i \, E_i[L(c)] \tag{2.18}$$

[64] argues that risk can never be considered in isolation, but must be considered together with the costs and benefits of the different possible actions. Furthermore, they argue that the risk of not acting must be considered on equal basis as the risk of different actions. The best action, a, to choose when considering expected risk is, therefore:

$$a = \arg\min_a \sum_i \int \phi p_i(\phi, a) d\phi \int L(c, a) \zeta_i(c, a) dc + C(a) \tag{2.19}$$

Where the action a can affect the probability of the scenario occurring, the probability of a loss occurring, and how bad the loss is ($L(c, a)$). $C(a)$ is the direct cost of performing action a. Benefits are here considered as positive consequences and are included in $L(c, a)$ as negative values.

Compared to the definitions given above, the definition of risk that is most commonly used in the fields of cybernetics, control theory, or artificial intelligence is very limited. Here it's common to state that a controller considers risk if it considers the probability and (sometimes) consequence of an unwanted event. In these fields, it is very uncommon to have the holistic perspective used in risk sciences, which is that we want to quantify all potential hazards, causal factors influencing the hazard, and losses caused by a hazardous event. Instead, when speaking of for example the risk of collision, it is common to implicitly consider only a single scenario of for example navigational uncertainty causing a collision.

In, Chapters 4 and 5, this thesis attempt to expand the definition of risk considered in automatic control to closer match the one presented in [64]. Chapter 4 uses the probability of occurrence as in Equation (2.8), while Chapter 5 uses the full probability distribution over frequencies definition as in Equation (2.9).

2.3 Decision making under uncertainty

This section is only needed by the curious reader who is interested in understanding the more general problem laying behind the problems solved in this thesis. This section is not needed to understand the rest of the thesis. The goal of this section is to give insight into how the problems tackled in this thesis relate to each other and give a motivation for why the heuristic approaches presented in this thesis are used instead of working with the general formulation.

First, a general formulation of decision-making under uncertainty is presented, this formulation is based on [48]. Thereafter some simplifications relevant to this thesis are presented followed by a comparison of the formulations given in this section and in Section 2.2. Lastly, a discussion on how the formulations given in the different chapters relate to the general solution is given.

2.3.1 General solution

The task of a robot is at a time step t to find an action a_t. To guide the robot in making a decision it has available all previous observations $O_{0:t} = o_0, o_1, ..., o_t$ and all previous actions that it has performed $A_{0:t-1} = a_0, a_1, ..., a_{t-1}$. In addition, to be able to interpret this information at all, it needs some prior information on how the world works, I. Let ρ denote the action policy that decides how the robot acts:

$$a_t = \rho(O_{0:t}, A_{0:t-1}, I) \tag{2.20}$$

It is common to use our prior knowledge to construct a dynamic model of how the world works. In particular, there are two models that are often used. The first is a model of how observations are made. As our knowledge about the world, I, is incomplete we are unable to model the observations with certainty. Instead, we employ a probabilistic model that states the probability of making an observation o_t when the world at time step t is in state $M_{i,t}$. Here subscript i denotes a particular state the world can be in. The observation model can be written as:

$$P(o_t|M_{i,t}) \tag{2.21}$$

Furthermore, we need a model of how the world reacts to our actions. Again a probabilistic model is needed. If the world is in state $M_{j:t-1}$ and we execute action a_{t-1} then what is the probability that the world will end up in state $M_{i:t}$? This transition model can be written as:

$$P(M_{i:t}|M_{j:t-1}, a_{t-1}) \tag{2.22}$$

Ideally $M_{i:t}$ should be a complete state of the universe at a particular time t. As this is impossible, we need to use our prior knowledge I to find a more compact representation of the state of the world. The representation of the world should be detailed enough to let us specify Equation (2.21) and Equation (2.22) with sufficient accuracy to meet our performance criteria. The formulation used in this section treats the world state as a discrete state that at any time is in one of a discrete set of possible world states, each identified with a unique index i. The formulation can be expanded to consider continuous state spaces by considering an uncountable set of world-states with infinitesimal differences between each discrete state.

With these models, we can use Bayes theorem to evaluate our belief that the world is in a particular state, $M_{i,t}$, given all past observations and actions.

$$P(M_{i,t}|O_{0:t}, A_{0:t-1}) = \eta P(o_t|M_{i,t}, O_{i:t-1}, A_{0:t-1}) P(M_{i,t}|O_{0:t-1}, A_{0:t-1}) \tag{2.23}$$

The same principle as in Equation (2.5) is used here with a normalization constant η to avoid all constant terms. If $M_{i,t}$ is a sufficient representation of the world then we do not need to consider past observations and past actions when modeling the current observation, this is often called the sensor Markov assumption [48]. The first term can therefore be simplified to $P(o_t|M_{i,t})$ which is our observation model, Equation (2.21). The second term can be evaluated using our transition model, Equation (2.22), by introducing the previous world state $M_{j,t-1}$.

$$P(M_{i,t}|O_{0:t}, A_{0:t-1}) = \eta P(o_t|M_{i,t}) \sum_j P(M_{i,t}|M_{j:t-1}, a_{t-1}) P(M_{j:t-1}|O_{0:t-1}, A_{0:t-2}) \tag{2.24}$$

We have here utilized that if $M_{j,t-t}$ is a sufficient representation of the previous state then we do not need to consider any earlier world states when evaluating the transition probability, this is often called a first-order Markov process [48]. Furthermore, we use the principle of causality which dictates that a state cannot be affected by actions performed at a future point in time.

With this, we have the formulas needed to recursively estimate the state of the world. The last piece we need to evaluate the current state is our prior belief regarding the world state before any observations or actions are made, that is $P(M_{j,0}, I)$.

Let the belief state, B_t represent the current belief that the world is in any of its possible states, $B_t = \{P(M_{i,t})\}$. The task of the robot is then to find what action to do given its current belief state:

$$a_t = \rho(B_t) \tag{2.25}$$

In many applications it is not enough to react based on our beliefs regarding the current state of the world, we also need to consider how we believe that the world will develop in the future. Equation (2.22) gives us the equation for how we believe the world will develop. As our knowledge of the world is incomplete, the uncertainty in the state of the world will typically increase as we predict forward in time. Directly using the transition model step by step into the future is overly pessimistic as it does not consider that we will make observations in the future that will reduce the uncertainty. The problem is that we do not know what these observations will be. Because of this uncertainty, we know that we will have a belief state in the future, but we are uncertain about what the belief state is. We, therefore, have to consider the probability of being in a belief state, $P(B_{k,t+1})$. The possible belief states are enumerated, each $B_{k,t+1}$ represents a particular belief state (probability of the world being in each possible state). As the probability of a particular world state, $P(M_{i,t})$, is a continuous variable, the belief state space must be represented as a continuous state space. Using the same principles as above, the probability of entering a belief state $B_{k,t+1}$ when we are in belief state $B_{l,t}$ and execute action $a_t$can be written as

$$P(B_{k,t+1}|B_{l,t},a_t)=\sum_{o_{t+1}}P(B_{k,t+1}|o_{t+1},B_{l,t},a_t)P(o_{t+1}|B_{l,t},a_t) \tag{2.26}$$

The first term either takes on the probability 1 or 0 as we deterministically know which belief state we will end up in if we know the prior belief state, prior action, and current observation. It evaluates to 1 if performing action a_t in belief state $B_{l,t}$ and then observing o_{t+1} places the robot in belief state $B_{k,t+1}$, and evaluates to 0 otherwise. The second term gives the probability of observing o_{t+1} when performing action a_t in belief state $B_{l,t}$. This term can be expanded to the following

$$\begin{aligned}P(o_{t+1}|B_{l,t},a_t)&=\sum_i P(o_{t+1}|M_{i,t},a_t)P(M_{i,t}|B_{l,t}) &(2.27)\\&=\sum_j\sum_i P(o_{t+1}|M_{j,t+1})P(M_{j,t+1}|a_t,M_{i,t})P(M_{i,t}|B_{l,t}) &(2.28)\end{aligned}$$

A common way of defining what action to execute in a belief state B_k is to define a reward, R, for being in state $M_{i,t}$, performing action a_t, and ending in state $M_{j,t+1}$.

$$R(M_{i,t},a_t,M_{j,t+1}) \tag{2.29}$$

As the true state is uncertain, we instead need to consider the expected reward for performing action a_t in belief state $B_{k,t}$.

$$R(B_{k,t},a_t)=\sum_i\sum_j R(M_{i,t},a_t,M_{j,t+1})P(M_{j,t+1}|M_{i,t},a_t)P(M_{i,t}|B_{k,t}) \tag{2.30}$$

To avoid making short-sighted decisions, the robot should not only consider the current reward but potential future rewards as well. To evaluate this we need to know how the robot will act in the future, however, as we do not know which observation that will be made, we cannot deterministically know which action that will be made in the future. A general solution to this problem is to define a policy ahead of time on how the robot should act in any possible belief state. The goal is then to choose the policy, ρ, that defines an action to perform in all possible belief state B_k, such that the utility, U, is minimized. The utility considers the direct expected reward and the indirect reward of continuing to follow this policy in the future.

There are multiple ways this utility can be defined, but a common way is to assume that it does not matter when the robot is in a particular belief state (it will always have the same optimal action in a particular belief state), and that rewards in the far future are less important than the current reward. The utility of each belief state can then be formulated as follows where γ is a discount factor discounting future rewards.

$$U(B_k|\rho) = R(a = \rho(B_k), B_k) + \gamma \sum_l U(B_{l,t}|\rho)P(B_l|\rho(B_k), B_k) \tag{2.31}$$

The optimal policy in a particular belief state is then one that gives the largest utility.

The formulation presented in this section on making decisions with uncertain measurements and transitions is called a partially observable Markov decision process (POMDP). For more information on this topic see [48], [65].

2.3.2 Simplifications

Even though this formulation describes the problem of acting in an uncertain world in an elegant manner, it will in practice be impossible to use in its general form for any but very small problems. The world state is often described using multiple properties that can be in different states. The total number of world states, therefore, grows exponentially with the number of properties and possible states they can be in. Furthermore, the number of belief states grows even quicker as it considers all possible combinations of beliefs we can have regarding the state of the world. Furthermore, even if the number of possible world-states that need to be considered can be radically reduced, it can still be prohibitively expensive to evaluate the optimal policy for all possible belief states ahead of time. The task of designing decision-making algorithms, therefore, consists of finding simplifications that makes it possible to solve the problem with limited computation and memory usage. This section will outline some common simplifications for representing the world and for designing policies that are relevant to the works presented in this thesis. As this is a very broad topic, only a very limited scope and depth are presented.

World representation

An approach for reducing the state-space explosion is to model the properties that make up the world-state using a DDN, or a DBN if we are only considering estimation. These models enable us to consider the probability distributions of the different properties separately from each other, as the network defines dependencies. This formulation does not lose any information compared to the original formulation, but has the trade-off that we can no longer only consider the state of different properties at the previous time step when evaluating their state at the current time step. This is caused by the dependencies between properties being a part of the model rather than being saved in the state. This means that we need to evaluate the entire history of actions and observations at each time step to get the current optimal estimate. This fact is hinted at in Section 2.1.3 where we had to repeat a BBN for each time-step to make a DDN or DBN. A common simplification is to consider a sliding window approach [56]. This approach considers a fixed set of time steps that are modeled in the DDN or DBN. The prior probabilities used on the first time step in the network are then gradually updated to be equal to the priors of the time steps no longer inside the window. This approach looses conditional information on states outside the window, but this can in many applications be an acceptable trade-off.

A common simplification used in control theory is to consider the expected world-state and its corresponding variance instead. This reduces the number of values that must be kept in mind to a vector of the expected value for the different properties of the world-state, and the covariance matrix between the properties. This formulation is famously used in the Kalman filter [66]. This only works under the assumption that all uncertainty can be modeled as additive Gaussian white noise and that the measurement and transition models are linear. Another alternative is to use the Monte Carlo sampling principle of particle filtering. Here multiple "particles" (world states) are considered and the ensemble of the particles represents the uncertainty in what the true world-state is [67].

Policy evaluation

In practice, it will for most problems be impossible to evaluate what to do in all possible belief states, even if the belief state is represented in a more compact form. This is typically handled by instead of finding the optimal solution ahead of time, the robot is tasked with finding an approximation of the best policy online. Online the robot knows which belief state it is in, and can therefore have a better idea of which future belief states are more likely to be relevant. It can therefore find an approximate solution that is pretty accurate around the current state by evaluating much fewer possible belief states. As this solution is accurate only around the current belief state it must be updated at regular intervals as the belief state changes. A survey of online planning algorithms for solving POMDPs is given in [68].

In control theory, model predictive control (MPC) [69] is commonly used to find the current action. Instead of finding the optimal policy, MPC commonly considers finding a finite set of future actions, an action sequence, $\{a_{t+1}, a_{t+2}, ..., a_{t+M}\}$. This formulation makes it significantly simpler to predict future world-states as we do not need to consider which observations that are made to know how our robot will act in the future. The most commonly used version of MPC does not explicitly consider uncertainty at all. This approach is feasible when the uncertainty in the current state and transition uncertainties is manageable through feedback and integral action. In this case, it can be sufficient to handle the uncertainty through feedback, that is, to see what happens and then correct for the errors caused by the uncertainty.

When the uncertainties are significant, then a stochastic MPC [69] can be employed. Stochastic MPC in its simplest form uses the same mean-variance formulation for the world state as in the Kalman filter. When working with an action sequence, the uncertainty regarding the future state will grow unrealistically fast. This is due to the prediction not considering that observations will be made in the future which gives us a better estimate that we will use to plan a new action sequence that compensates for any errors. Different approaches to alleviate this problem is presented in [70]. One of these methods is to apply a reactive controller that counteracts errors when predicting the future. This will not affect the expected future state but will limit how fast the variance grows. This controller must then be tuned to produce an approximation of how the variance would have grown if future observations were considered. In [70] different cost functions and constraints that are used to find the optimal action sequence are outlined. Note that constraints have not been mentioned in the general formulation as a constraint can always be represented by an infinitely negative reward for breaking the constraint.

An alternative to stochastic MPC is robust MPC [69]. Instead of modeling uncertainties as additive white noise, robust MPC assumes bounds on how large the uncertainties can be and then guarantee that constraints will be fulfilled even in worst-case conditions. One type of robust MPC strategy is the tube MPC [69]. Here bounded white noise is used to model the uncertainties in the transition model. With the bounded noise, a bounded set of possible future states can be evaluated. The world state is in the tube MPC represented by the outer limits of this set. Robust MPC has the same problem of the uncertainty growing unrealistically quickly as in the stochastic MPC case. Different possible solutions are presented in [69, ch.4].

A special case of the MPC formulation is the SB-MPC [14]. The idea behind the SB-MPC is to use a reactive lower-level controller that ensures that the robot achieves its goal quite well without any input from the SB-MPC. The SB-MPC will then modify the behavior of the lower-level controller by for example adding an offset to its proposed action or reference input. As the lower level controller takes care of most of the short-term considerations, the SB-MPC can consider a very short action sequence consisting of even only a single action. This action will then be applied for all future predicted time steps. The SB-MPC and the lower level controller often focus on different parts of the decision-making. The lower level controller can for example be tasked with correcting errors in how the robots pose develops over time relative to some reference, while the SB-MPC can focus on tasks such as avoiding collision. For avoiding collision the SB-MPC only needs to consider the environment, which in many cases is less affected by the choice of action than the state of the robot itself. This makes the problem of uncertainty in the future state growing unrealistically quick less of a problem than for stochastic or robust MPC. Furthermore, instead of considering a continuum of possible action the SB-MPC considers a finite set of possible action sequences. Considering a relatively small finite set of possible action sequences makes it computationally feasible to evaluate all action sequences which makes it possible to consider much more complicated cost functions and constraints than what is possible in ordinary MPC. This property makes it, among other things, possible to consider more complicated probabilistic models [71].

2.3.3 Decision-making under risk

Decision-making under uncertainty is inherently about risk as we have uncertainty in the consequence of actions. But there is a stark difference of the formulation given in Section 2.3 compared to Section 2.2.

First of Equation (2.19) considered only a single decision while Section 2.3.1 considered sequential decision making. This difference is due to the risk sciences often considering choosing between different system designs or safety precautions. The chosen design or precaution will often be held constant for a long time making it unnecessary to consider sequential decisions. This difference is not as large as it may first seem as the action a chosen in the design phase can be to choose a particular policy ρ the robot should use.

The second difference is how the problem is formulated. To compare the formulations, the expected risk formulation from Equation (2.12) considering a continuum of scenarios, and the online version of finding the optimal POMDP policy, are used. Changing the notation to match each other, the two formulations can conceptually be written as follows

$$\rho_{risk} = \arg\min_{\rho} \sum_{s} (L(s|\rho)P(s|\rho)) + C(\rho) \tag{2.32}$$

$$\rho_{POMDP} = \arg\min_{\rho} \sum_{t} (L(M_t|\rho)P(M_t|\rho, M_{t-1})) + C(\rho) \tag{2.33}$$

Here s is a particular scenario with a loss $L(s)$ and expected frequency of occurrence $P(s)$. The variable t represents time, and $L(M_t)$ evaluates the loss of being in the world-state M_t. $C(\rho)$ represents the cost of employing policy ρ. Rewards are considered to be negative losses.

From this, we can see that the risk formulation considers summing over scenarios while the POMDP formulation given in Section 2.3 considers summing over time. The risk formulation considers the expected loss of a scenario occuring together with the expected frequency, while the POMDP formulation considers the loss related to a world state and probability of being in that world state.

Using a dynamic model can be difficult in many accident analyses as the model has to be excessively complex to capture all the small nuances that affect whether an accident will occur. In these cases, it can be more useful to gather statistics on for example expected failure rates and aggregate this information to evaluate the frequency of a scenario occurring. A common way to aggregate information related to accident scenarios is by using a BBN, see for example [35]–[43].

2.3.4 Models used in this thesis

Chapter 3

This chapter considers the uncertainty in the location of obstacles and the robot itself. The world state is in this chapter described using two models, one for the location of obstacles and one for the location, pose, and velocity of the robot. The robot model uses the expected value and covariance formulation used in the Kalman filter. For the obstacles, the world is divided into cells that can be empty or occupied. The cells are then assumed independent to avoid the state-space explosion. This produces an occupancy grid map [72] where each cell has a probability of being occupied.

This chapter applies the SB-MPC formulation for deciding how the robot should move. The lower level controller employed by the SB-MPC is tasked with counteracting position errors caused by unmodelled disturbances affecting the robot and errors in the robot model, while the SB-MPC mainly focuses on avoiding collision. Only the state of the robot is considered when predicting forward in time. As the SB-MPC is not actively engaged in counteracting errors caused by uncertainty in this prediction, the unrealistic variance growth experienced in stochastic and robust MPC are avoided. The ability of SB-MPC to handle arbitrary cost and constraint functions enables us to consider the probability of collision at each time-step, considering both the navigational uncertainty of the robot and the uncertainty in the occupancy grid map.

Chapter 4

This chapter considers the effect of underlying faults with the robot and adverse environmental conditions that can prevent the robot from succeeding to execute different tasks. This chapter applied both the POMDP formulation and the risk formulation outlined in Section 2.3.3. The risk formulation is used to evaluate the outcome of a task execution attempt, while the POMDP formulation is used to model how the underlying faults develop over time and the effect of recovery actions and failed execution attempts on the state of the robot. A DDN is used to model both formulations in a combined model. The BBN at each time step of the DDN evaluates the outcome of task execution attempts, while the dynamics are used to model how the state of the robot develops over time. Using a DDN avoids the state space explosion as the world state is represented by different types of faults and environmental conditions.

The chapter simplified the problem of finding the optimal policy by creating three classes of heuristic policies consisting of one to three actions each. The action policies are here pre-defined similar to the case of MPC and are always re-planned after one action is executed. The unrealistic growth in uncertainty is handled by designing the cost function such that it considers the most relevant information, e.g. whether a task execution attempt is successful, and by formulating the problem in such a manner that a short enough time horizon can be considered making the transition uncertainties less important for decision making.

Chapter 5

Similarly to Chapter 4 this chapter also considers the effect of underlying faults and adverse environmental states. But instead of considering whether a task should be executed, it continuously considers what the different risk-influencing parameters shall be.

This chapter closely follows the risk formulation given in Section 2.2 for evaluating risk given the current estimate, and the formulation in Equation (2.24) for estimation. A BBN is used to evaluate, for a given parameter configuration and estimate of the state of causal factors, the frequency of different scenario categories occurring and the expected loss if a scenario occurs. Additionally, to infer the presence of different faults, the BBN is extended to a DDN. The choice of action is modeled as not influencing the underlying faults. This makes it possible to only consider the current state when evaluating the optimal action.

Chapter 6

This chapter considers the inference of the intentions of other ships at sea. As it only considers inference, it only considers the estimation problem. To avoid the state-space explosion a DBN is used similarly to the previous two chapters.

Chapter 7

This chapter uses the intention model developed in Chapter 6 in a collision avoidance algorithm. In addition to considering the uncertainties stemming from the intentions, this chapter also considers the navigational uncertainties of the own ship and target ships. Two models are used for the world-state. A DBN to model the intentions, and a mean-covariance formulation for the navigational uncertainty.

To find the optimal trajectories a SB-MPC formulation is used. A finite set of possible future own-ship and obstacle-ship trajectories are proposed. The intention model is used to evaluate the probability of the obstacle ship following any particular trajectory, whereas the mean-covariance formulation is used to model variation within a trajectory.

2.4 The International Regulations for Preventing Collisions at Sea (COLREGs)

This section is based on [52].

COLREGs [13] is divided into six parts (A to F) and has a total of 41 rules. Four Annexes detailing technical requirements for things such as ship lights and their shapes are also included. A simplified overview of the most relevant rules for collision avoidance from Part B on steering and sailing considering power-driven vessels is given below. See [13] for more information.

Part B - Steering and Sailing

Rule 7 *Risk of collision*: States that every vessel shall determine if there is a risk of collision, using all appropriate means and information from the current situation. It also states that a risk of collision exists if the compass bearing to the other vessel does not change significantly.

Rule 8 *Action to avoid collision*: Actions taken to avoid collision shall be made such that they are readily apparent for nearby vessels observing visually or by radar and be taken in ample time. This implies that large speed or course changes are preferred. Course changes should here be prioritized over speed changes for visibility when there is enough free space available.

Rule 13 *Overtaking*: A vessel is classified as overtaking if coming up to another vessel from a direction more than 22.5 degrees abaft its beam. If this is the case, then the overtaking vessel shall keep clear of the overtaken vessel.

Rule 14 *Head-on*: When two vessels meet on reciprocal or near reciprocal courses such that there is a risk of collision, then each vessel shall change their course to starboard such that they pass each other with the other vessel on the port side.

Rule 15 *Crossing*: When two vessels are crossing such that there is a risk of collision, then the vessel with the other on its starboard side shall keep out of the way, and avoid crossing ahead of the other vessel if possible.

Rule 16 *Action by give-way vessel*: The vessel supposed to give-way shall if possible perform substantial actions early to keep well clear of the other vessel.

Rule 17 *Action by stand-on vessel*: The vessel with a stand-on role shall nominally keep its course and speed, but should take action to avoid collision if the give-way vessel does not take appropriate action. Furthermore, if the situation considered is crossing, the stand-on vessel shall if possible not alter its course to port when the other vessel is on its port side. Having a stand-on role does not relieve a vessel from its give-way obligations.

Part I

Risk-based autonomy for inspection drones

Chapter 3

Risk-based obstacle avoidance

This chapter is based on the following publication

[46] **S. V. Rothmund** and T. A. Johansen, "Risk-Based Obstacle Avoidance in Unknown Environments using Scenario-Based Predictive Control for an Inspection Drone Equipped with Range Finding Sensors," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, Jun. 2019, pp. 221–230. DOI: 10.1109/ICUAS.2019.8797803

The method was developed by S. V. Rothmund with supervision from T. A. Johansen. Software development and simulations were done by S. V. Rothmund. The first draft was written by S. V. Rothmund and revised by T. A. Johansen

3.1 Introduction

3.1.1 Background and motivation

When moving close to static obstacles, such as for industrial inspection, including the uncertainties in the drone and obstacle positions are essential for safe and efficient operations. One approach to incorporate uncertainty in the position of the drone is to assume bounded noise and then to ensure that all possible positions where the drone can end up will not be in an obstacle. This is done for the linear case with linear constraints in [73] and for a nonlinear case with a predefined set of maneuvers in [74]. Another example of bounding is in [18] where the positional variance at each time-step is calculated and a constraint is introduced that requires that obstacles are k standard deviations away from the drone. Bounding the accepted uncertainty or accepted positional offsets makes sure that the probability of the drone colliding is smaller than the probability that the bounds were wrong. This method is conservative, which might be good when the goal is to get to the goal position without colliding with obstacles along the way. But when the goal is to fly close to objects for inspection, such conservative bounds can prevent the drone from getting as close to the object as is desired. Another limitation is that it does not give an obvious answer on what should be done when no action is feasible.

A less conservative approach would be to calculate the probability of collision and then put an upper bound on this probability. This probability could be set based on maximizing the revenue taking into account the value of the mission and the expected loss if a collision occurred. When no action fulfills the required probability for success, then the action that minimizes the probability of collision can be chosen. A method for model predictive control using an upper bound for the collision probability with linearly constrained obstacles is developed in [19]. This work assumes that the position and shape of the obstacles are known.

When the environment is not fully known, uncertainties in the environment must be considered to give a reasonable probability estimate of collision. One approach to describe obstacle uncertainty is to use occupancy grid maps. Each cell in these maps contains information on how likely it is that the current cell contains an obstacle. Mapping with range-finding sensors using occupancy grid maps was first introduced in [72]. This work lacked a computationally feasible method for updating the map for non-ideal sensor models. A linear time method that solved this problem was developed in [75].

Different methods for utilizing uncertainty in obstacle information with potential fields are shown in [76], [77]. [76] utilizes potential fields to push the vehicle further away from objects that have a higher certainty of existence, while [77] pushes the vehicle further away from objects where the positional uncertainty is larger. These works incorporate different aspects of uncertain environment information but do not incorporate the uncertainty in the position of the vehicle itself, and do not consider risk in a probabilistic sense.

[24] presents a method that incorporates both uncertainties in the position of the vehicle itself and mapping uncertainty. They use occupancy grid maps, but they do not consider the measurement uncertainty when constructing the map.

For small rotary-wing drones minimizing the weight of the payload is of great importance. With higher weight comes higher energy consumption which will reduce the operational time of a drone before it has to land and recharge. Enabling a drone to do collision avoidance with non-rotating body-fixed radar sensors instead of lidars would save a lot of weight, and thereby increase the operational time of the drone. Radars have the additional advantage of working in all weather and lighting conditions. A large drawback with radars is the large sensor uncertainty. This makes it essential to consider this uncertainty when building a probabilistic map of the environment.

3.1.2 Contribution

The main contribution of this paper is to develop a risk-based framework for obstacle avoidance in unknown environments that incorporates both uncertainty in the environment and the vehicle's position. The exact inverse sensor model proposed

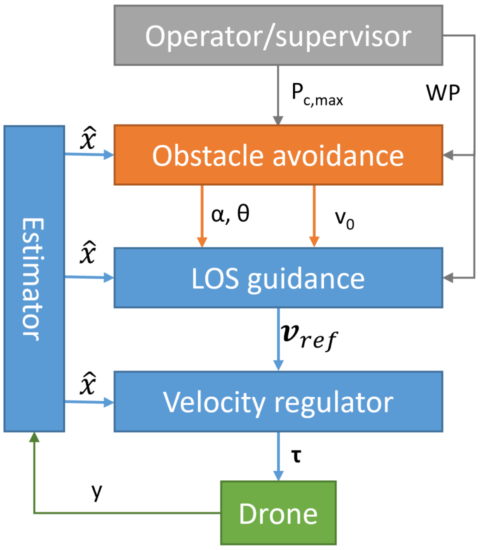


Figure 3.1: Control hierarchy. $P_{c,max}$ is the maximal accepted probability of collision at each time step. WP are the waypoints marking the planned path. α and θ are angle offsets relative to the nominal direction of motion. v_0 is the reference speed, $\mathbf{v}_{ref}$ the reference velocity, $\boldsymbol{\tau}$ are motor torques and forces, $\mathbf{y}$ are measurements, and $\hat{\mathbf{x}}$ is the estimated state.

in [75] is adapted for the case of wide-angle range finding sensors, such as radar, and is used to model the mapping uncertainty stemming from the measurement uncertainty. The SB-MPC developed in [14] for collision avoidance between ships is adapted to the inspection drone case and modified to use probabilistic uncertainty models that utilize uncertainties in the drone state and the environment to calculate the probability of collision. The collision avoidance strategy is generalized to 3D. To achieve this a 3D line-of-sight guidance strategy is proposed.

3.2 Overview

The goal of this chapter is to make an obstacle avoidance strategy for the execution of industrial drone inspections with an explicit awareness of the probability of collision with obstacles. The task of the drone is to collect data while following the straight lines between pre-planned waypoints. There might be unexpected obstacles blocking the planned path of the drone which forces the drone to deviate from the planned path to avoid collision. The drone is equipped with multiple wide-angle range finding sensors that give limited information about the obstacles. Examples of wide-angle range finding sensors are radar and sonar. The large field of view of the sensor and the position uncertainty of the drone at the time of sensing gives an uncertainty in the position and shape of detected obstacles.

The proposed control hierarchy is shown in Figure 3.1. The drone is controlled by a velocity controller that ensures that the drone moves in the designated direction at the designated speed. The velocity reference vector needed by the velocity controller is supplied by a line of sight (LOS) guidance law. This guidance law uses the list of waypoints to calculate the velocity reference that gently moves the drone toward and along the planned path.

The obstacle avoidance algorithm uses the SB-MPC formulation presented in [14] with a probabilistic uncertainty model that calculates the probability of collision over the prediction horizon. This method utilizes the LOS guidance to parameterize different paths using only one parameter in 2D and two parameters in 3D. This method has a limited set of possible control actions making it less complete than optimal control methods with full control over the drone's behavior, such as in [18]. The major advantage of the method is that the run-time is linear with respect to the number of possible combinations of control actions, and is easily parallelizable which makes it much faster than full optimal control solutions on multi-core processors. This makes the method feasible for real-time applications on weaker computational hardware, such as the ones present on drones. The method might be slower than potential field methods, but it avoids some of the inherent problems with a potential field such as unstable motion and getting stuck in potential minima close to narrowly spaced obstacles [78]. The proposed method also opens up for working with the probability of collision, which potential fields do not.

This obstacle avoidance algorithm gives a constant offset angle and reference speed to the LOS-guidance algorithm. The offset angle makes the drone gradually move away from the planned path specified by the waypoints. As the drone moves further away, the LOS guidance vector will point more directly toward the path, counteracting the offset. For offset angles under 90° the drone will converge toward a line parallel to the planned path. This behavior makes it possible to give a constant angle offset and still move in the along-path direction while executing an evasive maneuver. When the angle offset is set back to zero the LOS guidance law will automatically make the drone move back to the planned path made by the waypoints.

A finite set of angle offsets and velocity offsets are defined. A model of the drone system with LOS guidance and a velocity controller is simulated over a prediction horizon with all the different combinations of angle and velocity offsets. The control action is applied at the initial time step and kept constant over the entire prediction horizon. The probability of collision with the resulting behavior is checked against the maximal accepted probability of collision at each time step. The control action that maximizes the overall mission objective amongst the safe enough options is then chosen.

The probability of collision with the resulting behavior is calculated by combining a probability map over obstacle positions with a probability density function over the position of the drone. The probability map over obstacles is made online based on the range measurements from a radar or sonar. The drone's position is not exactly known at the current time step due to uncertainties in the sensors used for estimation. When predicting into the future, the position of the drone gets less certain over time as some unknown disturbance might affect it. The drone will have controllers that will counteract these errors, but these controllers have dynamics that make them unable to instantly counteract disturbances.

3.3 Drone and control model

3.3.1 Open loop model

The drone is for simplicity assumed to be a fully actuated double integrator, driven by an acceleration caused by the control input $\mathbf{u}$, and affected by an additive disturbance $\mathbf{w}_c$. The state is written on the form $\mathbf{x} = [\mathbf{x}_p^\top, \dot{\mathbf{x}}_p^\top]$, where $\mathbf{x}_p$ is the position of the drone decomposed in a north east down (NED) coordinate frame.

$$\dot{\mathbf{x}} = A_c\mathbf{x} + B_c\mathbf{u} + \mathbf{w}_c \tag{3.1}$$

$$A_c = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad B_c = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \tag{3.2}$$

As this model is linear, an exact discretization for A_c and B_c can be found, these are denoted as A and B. The discrete-time white noise process $\mathbf{w}$ is assumed to be Gaussian with a covariance matrix denoted as Q. The notation $\mathbf{x}[k] = \mathbf{x}(k\ dt)$ is used, where dt is the discretization time step. To simplify notation the time-step index is only included in the state update equations.

$$\mathbf{x}[k+1] = A\mathbf{x}[k] + B\mathbf{u}[k] + \mathbf{w}[k] \tag{3.3}$$

3.3.2 Velocity controller

The drone is equipped with a velocity controller.

$$\mathbf{u}[k] = -K(\begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \hat{\mathbf{x}}[k] - \mathbf{v}_{ref}[k]) \tag{3.4}$$

$$\hat{\mathbf{x}}[k] = \mathbf{x}[k] + \mathbf{v}[k] \tag{3.5}$$

Where K is a gain matrix and $\mathbf{v}_{ref}$ is the reference velocity vector. This controller uses the estimated state, $\hat{\mathbf{x}}$, which is modeled as the true state, $\mathbf{x}$, plus some estimation error, $\mathbf{v}$, that is assumed to be a discrete Gaussian white noise process with zero mean and covariance matrix R.

The closed-loop dynamics is then given as follows

$$\mathbf{x}[k+1] = A_{cl}\mathbf{x}[k] + B_{cl}\mathbf{v}_{ref}[k] - \Gamma_{cl}\mathbf{v}[k] + \mathbf{w}[k] \tag{3.6}$$

$$A_{cl} = A - BK\begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix}, \quad B_{cl} = BK, \quad \Gamma_{cl} = BK\begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3.7}$$

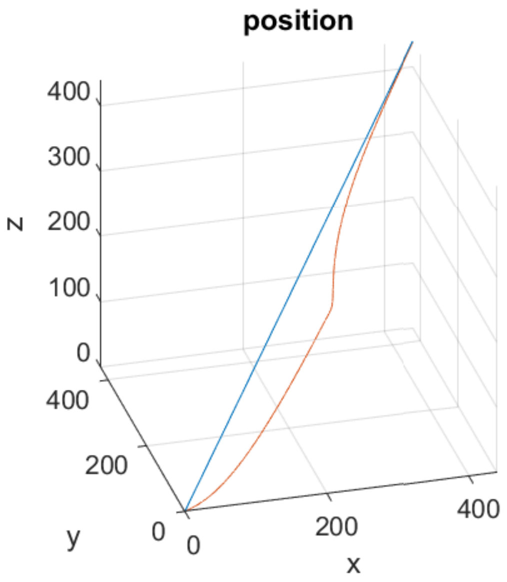


Figure 3.2: LOS guidance in 3D. Blue marks the nominal path while orange marks the path followed by the drone. An offset $\alpha = 40°$ in direction $\theta = 160°$ is applied in the first half of the simulation, then the offset is turned off and the drone returns to the nominal path.

3.3.3 Line of Sight guidance

A 3D line-of-sight guidance strategy is needed. Four different options are compared in [79]. These methods produce reference yaw and pitch angles. A method that instead produces a reference velocity vector is presented here. This method avoids trigonometric functions which simplifies the linearization of the resulting dynamics.

To formulate the LOS guidance law in 3D an additional coordinate system, called the LOS coordinate system, is defined. This coordinate system is defined as having the x-axis along the line connecting the previous and the next waypoint, denoted as WP_1 and WP_2. The y- and z-axis can be arbitrarily chosen as long as the LOS coordinate system is a right-hand coordinate system. $\mathbf{y}_{LOS}$ is chosen to be the cross product between $\mathbf{x}_{LOS}$ and the z-axis in the NED frame.

$$\mathbf{x}_{LOS} = \frac{WP_2 - WP_1}{||WP_2 - WP_1||} \tag{3.8}$$

$$\mathbf{y}_{LOS} = \frac{\mathbf{x}_{LOS} \times \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top}{||\mathbf{x}_{LOS} \times \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top||} \tag{3.9}$$

$$\mathbf{z}_{LOS} = \mathbf{x}_{LOS} \times \mathbf{y}_{LOS} \tag{3.10}$$

The position of WP_1 and WP_2 as well as the vectors $\mathbf{x}_{LOS}$, $\mathbf{y}_{LOS}$, and $\mathbf{z}_{LOS}$ are given in the NED frame.

For the special case where $\mathbf{x}_{LOS} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^\top$, where the cross product in Equation (3.9) is undefined, the alternative formulation is used.

$$\mathbf{z}_{LOS} = \frac{\mathbf{x}_{LOS} \times \begin{bmatrix}0 & 1 & 0\end{bmatrix}^\top}{||\mathbf{x}_{LOS} \times \begin{bmatrix}0 & 1 & 0\end{bmatrix}^\top||} \tag{3.11}$$

$$\mathbf{y}_{LOS} = \mathbf{z}_{LOS} \times \mathbf{x}_{LOS} \tag{3.12}$$

This basis can be used to find the rotational matrix between NED and LOS.

$$R_{LOS}^{NED} = \begin{bmatrix}\mathbf{x}_{LOS} & \mathbf{y}_{LOS} & \mathbf{z}_{LOS}\end{bmatrix} \tag{3.13}$$

The difference between the drone's position and WP_1 given in the LOS frame gives the drone's distance along and offset from the path between the two waypoints. The x coordinate is the distance along the path, while the y and z coordinates give the offset across the path. LOS guidance makes the drone at all times follow the vector pointing from its current position to a point Δ ahead on the planned path. In the LOS frame, this vector has coordinate Δ in x_{LOS} direction, and the y and z components of the distance from the drone to WP_1 in the y_{LOS} and z_{LOS} direction. By normalizing this vector and multiplying it with the desired speed, v_0, the reference speed vector that the drone should follow is generated:

$$\chi_{LOS}^{NED} = R_{LOS}^{NED}\left(\begin{bmatrix}\Delta \\ 0 \\ 0\end{bmatrix} - \begin{bmatrix}0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1\end{bmatrix} R_{NED}^{LOS}(\begin{bmatrix}\mathbf{I} & \mathbf{0}\end{bmatrix}\hat{\mathbf{x}} - WP_1)\right) \tag{3.14}$$

$$\mathbf{v}_{ref} = v_0 \frac{\chi_{LOS}^{NED}}{||\chi_{LOS}^{NED}||} \tag{3.15}$$

The estimated drone state, $\hat{\mathbf{x}}$, is given in the NED frame. Note that the line of sight guidance system makes decisions based on the current best estimate of the state, $\hat{\mathbf{x}}$, not the actual state $\mathbf{x}$.

The obstacle avoidance controller introduces an offset angle to the velocity vector. In the 2D case developed in [14], an angle α is added to the LOS angle. When seen as a vector, this is the same as rotating the vector by α about the z-axis pointing out of the plane. For the 3D case, two parameters are needed, α and θ. α is used for rotating the vector around some axis orthogonal to the $\mathbf{x}_{LOS}$ axis. The angle θ tells us the orientation of this axis relative to the $\mathbf{y}_{LOS}$ axis. This is done using the rotation matrix shown in Equation (3.17).

$$\begin{aligned}
&\chi_{LOS,ca}^{NED} = \\
&R_{LOS}^{NED} R_{ca} \left(\begin{bmatrix} \Delta \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} R_{NED}^{LOS} (\begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \hat{\mathbf{x}} - WP_1) \right) && (3.16) \\
&R_{ca} = R_{x=-\theta} R_{y=\alpha} R_{x=\theta} && (3.17) \\
&\mathbf{v}_{ref,ca} = v_0 \frac{\chi_{LOS,ca}^{NED}}{||\chi_{LOS,ca}^{NED}||} && (3.18)
\end{aligned}$$

The resulting behavior with a constant offset angle is shown in Figure 3.2. This figure also shows that the drone gently moves back to the path when α is set to zero. How quickly the drone should move towards and away from the path is specified by Δ.

Special 2D case

The vector-based 3D line of sight formulation can easily be used in 2D as well but requires some special notation as the cross product and rotational matrices are not defined for 2D.

$$\mathbf{x}_{LOS} = \frac{WP_2 - WP_1}{||WP_2 - WP_1||} \tag{3.19}$$

$$\mathbf{y}_{LOS} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} x_{LOS} \tag{3.20}$$

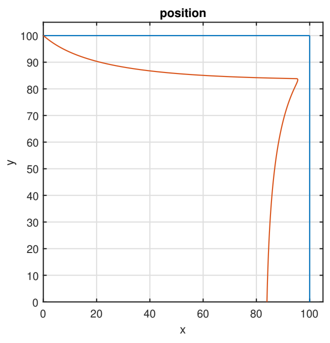
$$R_{LOS}^{NED} = \begin{bmatrix} \mathbf{x}_{LOS} & \mathbf{y}_{LOS} \end{bmatrix} \tag{3.21}$$

$$\chi_{LOS,ca}^{NED} = R_{LOS}^{NED} R_{ca} \left(\begin{bmatrix} \Delta \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} R_{NED}^{LOS} (\begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \hat{\mathbf{x}} - WP_1) \right) \tag{3.22}$$

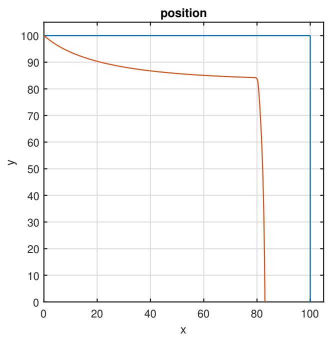
$$R_{ca} = \begin{bmatrix} \cos(\alpha) & -\sin\alpha \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{3.23}$$

Switching between waypoints

Two common ways of switching waypoints in line-of-sight guidance are presented in [80]. One approach is to change waypoint when the vehicle is within a given radius of the current waypoint (circle of acceptance). As the anti-collision control actions might take us far away from the waypoints, this might lead to the waypoint being missed. The other strategy is to change waypoint when the along-path distance to the next waypoint is small enough. This strategy works well when the goal is to follow the desired path closely but leads to unwanted behavior when there is a wanted offset due to the control action made by the obstacle avoidance system. The drone might then switch waypoint closer to the next path segment than the designed offset. This is shown in Figure 3.3(a).



(a) Unwanted behavior when switching waypoints based on along-path distance.



(b) Correct behavior when switching based on the relative distance to the two lines.

Figure 3.3: Behaviour with different strategies for changing waypoints. Orange shows the drone position with a constant angle offset and blue marks the nominal path between the waypoints. The drone starts at $x = 0$, $y = 100$.

This can be solved by switching waypoints when the drone is closer to the next path segment than to the current path segment. This will avoid making the drone move closer and then further away from the path segment. A margin can be implemented to compensate for the drone dynamics, making the drone switch waypoint a bit before it's equally close to both path segments. The distance to the path segment should be the closest distance to any point on the infinite line going through the waypoints. The shortest distance a point $\mathbf{x}$ is away from the infinite line going through points $\mathbf{a}$ and $\mathbf{b}$ can be calculated as

$$s(\mathbf{a}, \mathbf{b}, \mathbf{x}) = (\mathbf{x} - \mathbf{a})^\top \frac{(\mathbf{b} - \mathbf{a})}{||\mathbf{b} - \mathbf{a}||} \tag{3.24}$$

The behavior of this waypoint switching algorithm is shown in Figure 3.3(b).

3.4 Heading dynamics

For the fully actuated double integrator drone model, the heading does not affect the position and velocity dynamics, as the drone is able to fly in any direction independent of the heading. But as both the payload sensors (e.g. camera) and range-finding sensors may be predominantly placed in one direction, the drone may have to turn the sensors towards the movement direction to be able to detect obstacles in its way. A simple heading dynamic is implemented in the simulator to include this behavior.

$$\psi[k+1] = \gamma\psi[k] + (1-\gamma)\psi_{ref}[k] \tag{3.25}$$

$$\psi_{ref} = \text{atan2}(V_{ref,ca,x},\, V_{ref,ca,y}); \tag{3.26}$$

Where ψ denotes the heading. The parameter $\gamma \in < 0, 1 >$ decides how quick the heading dynamics will be.

3.5 Covariance propagation

3.5.1 Covaraince formulation

The LOS guidance law is nonlinear due to the normalization of the $\chi_{los,ca}^{NED}$ vector in Equation (3.18). Nonlinearities will distort a Gaussian probability distribution making it difficult to propagate the covariance. The system is linearized to avoid this problem.

First, the guidance law Equation (3.16) is re-written.

$$\chi_{LOS,ca}^{NED} = E - F\hat{\mathbf{x}}_p \tag{3.27}$$

$$= E - F\mathbf{x}_p - F\mathbf{v}_p \tag{3.28}$$

$$E = R_{LOS}^{NED} R_{ca} \left(\begin{bmatrix} \Delta \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} R_{NED}^{LOS} W P_1 \right) \tag{3.29}$$

$$F = R_{LOS}^{NED} R_{ca} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} R_{NED}^{LOS} \tag{3.30}$$

$$\mathbf{x}_p = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}, \quad \hat{\mathbf{x}}_p = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \hat{\mathbf{x}}, \quad \mathbf{v}_p = \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{v} \tag{3.31}$$

Both $\mathbf{x}$ and $\mathbf{v}$ are stochastic variables where $\mathbf{x}$ is the state and $\mathbf{v}$ describes the uncertainty due to measurement errors. Inserting Equation (3.28) into Equation (3.18) yields

$$\mathbf{v}_{ref,ca} = v_0 \frac{E - F\mathbf{x}_p - F\mathbf{v}_p}{||E - F\mathbf{x}_p - F\mathbf{v}_p||} \tag{3.32}$$

This equation is linearized around the estimated expected position evaluated at time-step k denoted as $\mathbf{x}_k$. This state should be propagated through the closed loop state space Equation (3.6) using the nonlinear LOS guidance Equation (3.18) for the velocity reference vector. The linearization is done around $\mathbf{v} = 0$ as $\mathbf{v}$ is assumed to have zero mean. The linearization results in the following equations

$$\bar{\mathbf{v}}_{ref,ca} = v_0(\bar{E} - \bar{F}\mathbf{x} - \bar{F}\mathbf{v}) \tag{3.33}$$

$$\bar{E} = G + H \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k \tag{3.34}$$

$$\bar{F} = H \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \tag{3.35}$$

$$G = \frac{E - F \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k}{||E - F \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k||} \tag{3.36}$$

$$H = \frac{F}{||E - F \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k||} - \left(\frac{(E - F \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k)(E - F \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k)^\top F}{||E - F \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{x}_k||^3} \right) \tag{3.37}$$

Inserting the linearized velocity reference vector into the state Equation (3.6) leads to

$$\mathbf{x}[k+1] = A_{LOS}\mathbf{x}[k] + \Gamma_{LOS}\mathbf{v}[k] + \mathbf{w}[k] + C_{LOS} \tag{3.38}$$

$$A_{LOS} = A_{cl} - B_{cl}v_0\bar{F}, \quad \Gamma_{LOS} = -\Gamma_{cl} - B_{cl}v_0\bar{F} \tag{3.39}$$

$$C_{LOS} = B_{cl}v_0\bar{E} \tag{3.40}$$

We now have a linear state space formulation. With the assumption that $\mathbf{v}[k]$ and $\mathbf{w}[k]$ are independent white noise processes, all the input terms in Equation (3.38) are uncorrelated and the covariance matrix of $\mathbf{x}[k+1]$ can be calculated as

$$\text{var}(\mathbf{x}[k+1]) = A_{LOS}\text{var}(\mathbf{x}[k])A_{LOS}^\top + \Gamma\text{var}(\mathbf{v}[k])\Gamma^\top + \text{var}(\mathbf{w}[k]) \tag{3.41}$$

$$\text{var}(\mathbf{x}[k+1]) = A_{LOS}\text{var}(\mathbf{x}[k])A_{LOS}^\top + \Gamma R\Gamma^\top + Q \tag{3.42}$$

The initial variance is equal to the state estimator variance, R.

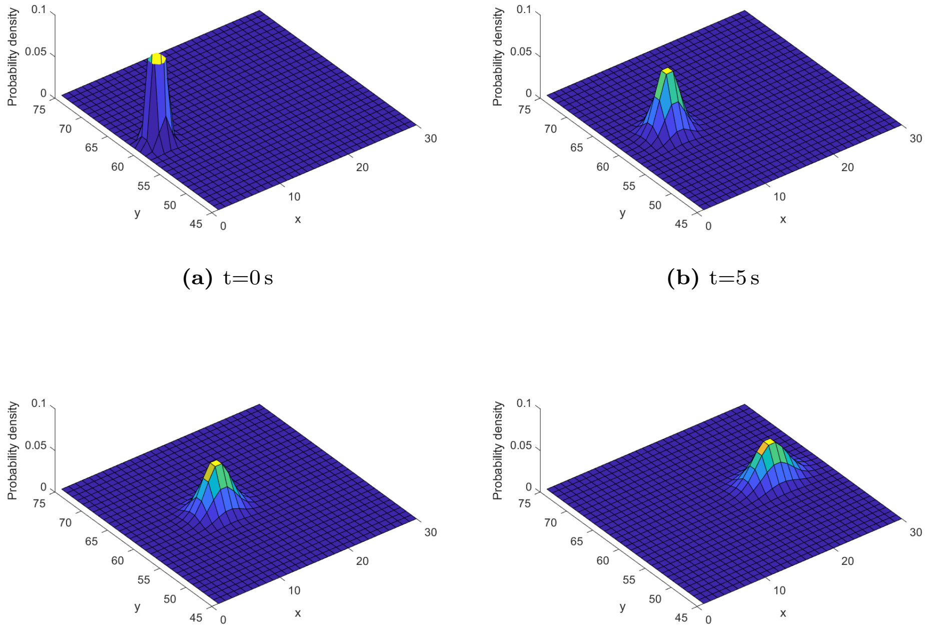
$$\text{var}(\mathbf{x}[0]) = R \tag{3.43}$$

3.5.2 Resulting covariance dynamics

Figure 3.4 shows how the uncertainty in position varies over time in the prediction. The figure shows that the uncertainty in the predicted position will start low and then gradually increase. It is interesting to note that the probability density function becomes elongated over time, having a larger uncertainty in the along-path direction than in the across-path direction. This is a direct consequence of LOS guidance only counteracting across path error, making it asymptotically stable in across path direction but only marginally stable in the along-path direction.

3.6 Mapping

To be able to avoid obstacles, a map has to be made based on the data from range-finding sensors. This work assumes that one or more body-fixed sonars or



(a) t=0 s **(b)** t=5 s **(c)** t=10 s **(d)** t=20 s

Figure 3.4: The probability density function of the drone's predicted position at different time steps into the future.

radars are used. A laser range finder would be a special case where the field of view of the range-finding sensor is just one line. The sensor is assumed to return the shortest distance to any object within a cone with a width equal to the field of view of the sensor. A radar would return multiple reflections, but only the first is used as it is uncertain whether later reflections are caused by the radar wave leaving or entering a new material. The exact location of the object inside the field of view is unknown, only the distance from the sensor and the fact that it is inside the field of view is known. This model is quite simplistic and does not incorporate specular reflections or multipath. Specular reflection is when the entirety of the emitted signal is reflected away from the sensor, which will not give a distance measurement. Multipath is when the signal is bounced off multiple surfaces before returning to the sensor, which makes the measured distance longer than the true distance to the target. One method for handling specular reflections is presented in [81]. There will be uncertainties in the position of a measured obstacle as there will be uncertainties in the range measurement and in the position of the drone at the time of the measurement. The uncertainties are assumed to be Gaussian. The variance in position in the direction of the measurement is added together with the variance of the sensor output to give the measurement uncertainty, σ_t^2.

The exact solution when using occupancy grid maps is to assign probabilities to map states m_i where each cell is either occupied or not. When a measurement r is made, the probability of the different map states being the correct can be updated as follows:

$$P(m_i|r) = \frac{p(r|m_i)P(m_i)}{p(r)} \tag{3.44}$$

Here $P(m_i)$ represents the prior probability of this map state being true and $p(r|m_i)$ represents the inverse sensor model.

The probability density of making a measurement r in map state m_i can be evaluated by finding the closest distance to an occupied cell within the sensor cone of map state m_i called $m_{i,d}$, and then evaluating the following normal probability density function:

$$p(r|m_i) = \mathit{normalpdf}(x = r, \mu = m_{i,d}, \sigma = \sigma_t) \tag{3.45}$$

As the number of possible map states grows exponentially with the number of states it is impossible to exactly evaluate the map probabilities. Instead, as proposed by [72], each cell is considered independent making it possible to consider the probability of a cell, c_i being occupied, $P(c_i)$, rather than the probability of the map state, m_i being true. A recursive method for updating the occupancy probabilities while considering sensor uncertainty was developed by [75]. This method was developed for a laser sensor where we get a 1D line of cells that could be in the path of the sensor. This method can be extended to the 2D case of a range-finding sensor with a larger field of view by first finding all cells within the field of view and then sorting them based on their distance to the sensor. This will produce a 1D array of cells, where cells with lower indexes block cells with higher indexes as they are closer to the sensor. This 1D array can be directly inserted into the method of [75]. Figure 3.5 shows how the map looks like after one update with a measured distance of 20 meters with $\sigma_t^2 = 1m^2$ and the grid cell size of 1x1 meters.

Measurements from different sensors and at different time steps can be incorporated by examining them one at a time. The resulting map from one update should be used as the a-prior map for the next sensor update. The map is initialized to each cell having a uniform chance of containing an obstacle. If a prior map over the environment is known, then the map could instead be initialized with a higher value where obstacles are expected to be.

3.7 SB-MPC formulation

The SB-MPC strategy will compare different control actions and choose the action that maximizes the mission objective while having an acceptable probability of collision. For the 2D case, the list of candidate control actions is defined as follows.

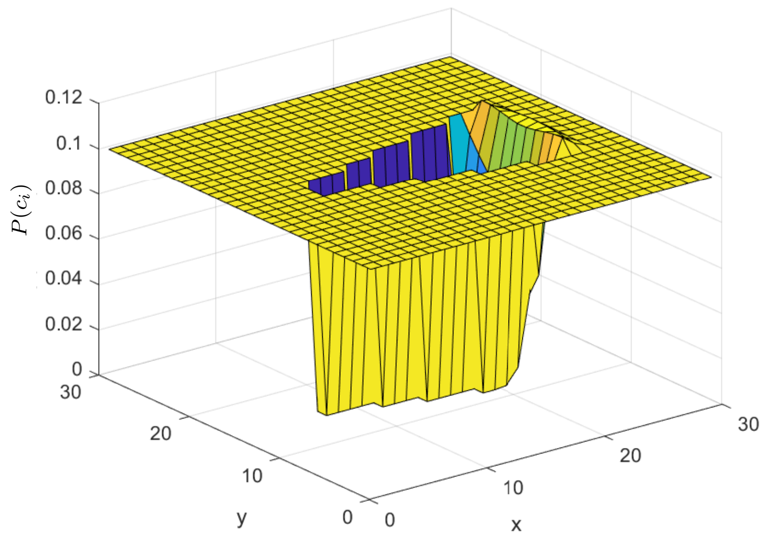


Figure 3.5: Occupancy grid map updated with one measurement of a wide-angle range-finding sensor. The z-axis gives the probability of there being an obstacle in a particular cell, $P(c_i)$. The prior probability of obstacles was set to 0.1 in all cells.

$$\begin{aligned} \alpha = [\ & -90 \quad -75 \quad -60 \quad -45 \quad -30 \quad \cdots \\ & -15 \quad 0 \quad 15 \quad 30 \quad 45 \quad 60 \quad 75 \quad 90\ ] \end{aligned} \tag{3.46}$$

$$v_0 = \begin{bmatrix} v_0^* & 0.5v_0^* & 0.25v_0^* \end{bmatrix} \tag{3.47}$$

Where v_0^* is the nominal speed of the drone. The resulting possible trajectories from a stationary start-point along a straight path going downwards are shown in Figure 3.6.

The drone model will be used to predict future behavior when applying the different combinations of angle and velocity control actions. The predicted state, as well as the variance in the estimate at each future time step is used to check the constraint and calculate the cost. The control action that optimizes the objective (see Section 3.7.2) is chosen among the feasible actions that fulfill the constraints (see Section 3.7.1). If no action is feasible then a default action will be taken (see Section 3.7.3). The procedure is repeated at regular sampling intervals.

Only essential constraints and objectives are implemented to highlight how this algorithm works. Other objectives could be added and tuned to give better mission-specific performance.

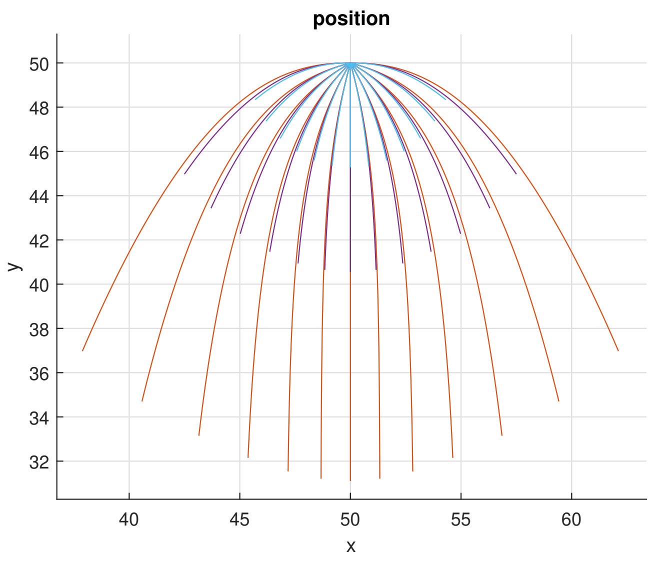


Figure 3.6: All candidate trajectories from a stationary initial position where the nominal path goes straight down. The different colors represent different speeds set points, orange represents $v_0 = v_0^*$, purple $v_0 = 0.5v_0^*$, and blue $v_0 = 0.25v_0^*$.

3.7.1 Constraint - Probability of collision

To ensure that the chosen route is safe, the probability of collision, P_c, has to be lower than a specified maximum collision probability, $P_{c,max}$, for all time steps, t in the simulation time-horizon, T_h. The time horizon is chosen equal to the time needed to identify obstacles, re-plan, and execute an evasive maneuver, thereby ensuring that the probability of collision is acceptable over the reaction time of the obstacle avoidance algorithm. With t_0 indicating the current time-step the constraint can be formulated as follows:

$$\forall_{t\in\{t_0,t_0+dt,\ldots,T_h}\}P_c[t] < P_{c,max} \tag{3.48}$$

The probability of collision at a specific time-step, $P_c[t]$, is equal the probability of the drone being in the same cell as an obstacle. This is calculated by summing over all cells, c_i, the probability of that cell containing an obstacle, $P(c_i)$, times the probability that the drone is within that cell at that time-step $P(D_i[t])$. With N indicating the number of cells this can be formulated as follows:

$$P_c[t] = \sum_{i=1}^{N} P(c_i)P(D_i[t]) \tag{3.49}$$

The probability of the drone being within a cell at time-step t, $P(D_i[t])$, is evaluated

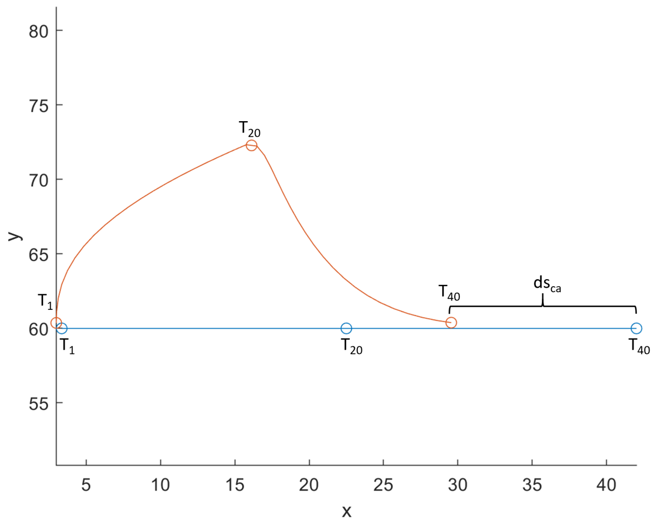


Figure 3.7: Red marks the path followed by the drone if a control action of $\alpha = 90°$ is set for the first 20 time-steps, and then turned off for the next 20. Blue marks the corresponding path with the nominal control action. ds_{ca} marks the resulting reduction in traversed distance.

by integrating the probability distribution of the location of the drone at that time-step over the area of the cell. The probability distribution of the drone's location is described as a normal distribution with the mean and variance given by Equations (3.38) and (3.42).

3.7.2 Objective

The constraint ensures that the chosen path is safe. The objective function can now be freely chosen based on the objective of the mission. One objective that ensures progress along the path must be implemented. This can be done by maximizing the traversed distance along the path. Deviations make the drone travel orthogonal to the path reducing the traversed along-path distance. The drone must fly back to the path at some point which introduces further delays. This is illustrated in Figure 3.7. To figure out the delay introduced by a control action, the drone is first simulated as normal with the control action active and is then further simulated with the nominal control action until it reaches back to the path. The nominal action is the behavior with $\alpha = 0$ and $v_0 = v_0^*$. This second simulation is done without variance propagation and checking the risk constraint. This will significantly speed up the second predictive simulation. By simulating the behavior from the control action plus the behavior on returning to the path, the delay introduced by the control action is captured. The traversed along-path distance is compared to the case where only the nominal action is applied. The method is described in Algorithm 1 and the resulting predicted reduction in along-path distance, ds_{ca}, introduced by the control actions is minimized.

The prediction should stop when the drone is sufficiently close to the nominal path. The acceptable cross-track error is denoted as δ. This has to be done as the drone will asymptotically move towards the nominal path, but may never completely hit it. With δ small, the resulting loss in along-path distance is negligible. The larger δ is, the quicker the simulation is finished. A trade-off between computational time and precision has to be made.

Algorithm 1 Calculation of relative distance along the path.

```
Let $S_0(t)$ denote the along-path distance for the nominal action at time $t$.
Let $T_{0,max}$ denote the latest time that is predicted for the nominal action.
Let $S_{ca}(t)$ denote the traversed distance for control action (ca) at time $t$.
Let $T_{ca,max}$ denote the latest time that is predicted for the control action ca.
Let $T_h$ be the time until a new control action will be chosen by the obstacle
avoidance algorithm.
Simulate $S_0(t)$ for $t = 0$ to $t = T_h$
for all control actions $ca$ do
    Simulate $S_{ca}(t)$ for $t = 0$ to $t = T_h$
    while cross track error at $T_{ca,max} > \delta$ do
        Simulate $S_{ca}(t)$ for $t = T_{ca,max} + 1$ without variance propagation
    end while
    if $T_{0,max} < T_{ca,max}$ then
        Simulate $S_0(t)$ for $t = T_{0,max}$ to $t = T_{ca,max}$ without variance propagation
    end if
    $ds_{ca} \leftarrow S_0(T_{ca,max}) - S_{ca}(T_{ca,max})$
end for
```

3.7.3 Default action

When no action is feasible then the safest action among all the given actions and the stop action should be taken. Often when the drone gets stuck, stopping might be the safest choice. But if measurement errors lead to the current drone position being dangerous, then it might be safer to take a non-zero action that will move the drone further away from obstacles. If the stop action is chosen then the drone should start rotating to improve the obstacle map.

3.8 Simulation

The simulations were done with 5 sensors, pointing forward, 30 degrees to the side, and 60 degrees to the side. The field of view of the sensors was set to $\pm 45°$ and the range set to 30 meters. The position and velocity variance returned from the estimator, R, and the model variance, Q, were set to $0.1m^2$. The variance in the measured range is set to $0.5m^2$. The accepted probability of collision per time-step is set to 0.1%. The map is initialized with a 10% a priori probability of containing an obstacle. The parameters were set to give interesting behavior that highlights the workings of the algorithm.

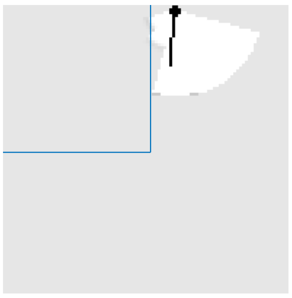
Figure 3.8 shows that the drone successfully manages to safely fly around a corner and into a tight corridor. An object is present in the blind spot around the corner. The drone chooses to take a slight detour to acquire new information about the corridor before it flies into it. This ensures that it avoids the obstacle. In the corridor, the drone reduces its speed as the wide field of view of the range-finding sensors makes it hard to distinguish walls from the safe area in between. Reducing the speed gives the drone more time to acquire new data and make a new plan.

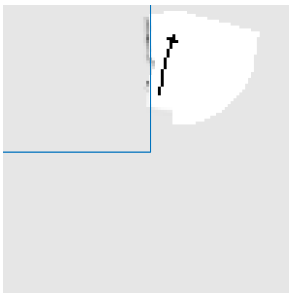
Figure 3.9 shows that the drone manages to find its way through a narrow opening. Figure 3.10 shows the limitations of this algorithm. Figure 3.10(a) shows that the drone manages to fly around smaller obstacles, but unable to circumvent larger obstacles as in Figure 3.10(b). This is caused by there not being a control action that lets the drone fly in a large enough arch with the given dynamics and look-ahead distance Δ. Figure 3.10(c) shows that the drone can get stuck in convex hulls without being able to escape. A solution to this problem would be to include more extensive control action candidates and to have a higher level controller that detects that the drone is stuck and re-plans the path taking the new obstacle information into account.

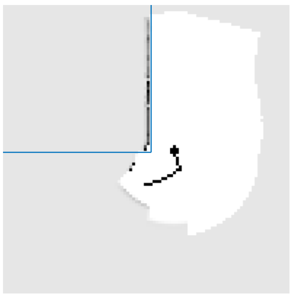
3.9 Discussion and conclusion

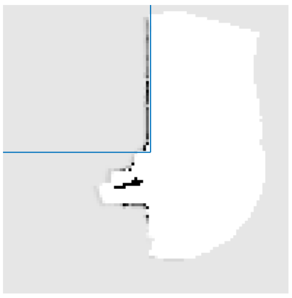
This chapter has looked into obstacle avoidance based on the probability of collision and developed an obstacle avoidance strategy that ensures that the probability of collision is at an acceptable level for all time steps. The essential constraint, which is that the path is safe over the critical time needed to re-plan and stop, is implemented. The essential objective which is to traverse the path is implemented as well. The simulation study showed that the proposed rules ensured that the drone was safe at all times and that the drone managed to avoid smaller obstacles. The strategy worked with the limited mapping capabilities of range-finding sensors with a wide field of view. It produced the behavior of looking around corners before entering and flying slower when only information about a limited area is known. The strategy forced the drone further away from the walls when the planned path incurred too much risk. The strategy is greedy, making it in some cases unable to find a path around larger obstacles and out of convex hulls. The proposed strategy ensures that the drone will be safe for all paths, and will stop or move to a safer point when no path is feasible. This enables a higher-level controller, or human planner, to plan a path based on a simplified map without taking the safety or dynamics of the drone into consideration.

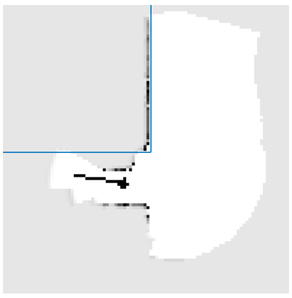
As we want to limit the actual probability of collision, the assumptions in the mapping method must be discussed. The main assumption in an occupancy grid map is that all the cells are independent. This assumption does not hold as all the updated cells from one measurement will be dependent as they give information on where one object is located. If it turns out that the object is not in one cell, then the probability that it is in another cell is increased as the object has to be somewhere. For different measurements of different objects, the independent assumption holds.


(a) t=0 s. The initial plan of the drone. The size of the sensor cone makes it impossible to determine if flying toward the path is safe.


(b) t=12 s. After mapping for a few seconds, a better map of the world is achieved. It is now deemed safe to fly a bit closer to the planned path.


(c) t=51 s. The drone is tasked to fly around the corner. To avoid flying into unknown territory the drone makes a larger turn.


(d) t=92 s. The drone enters a narrow corridor where the sensors' large field of view makes it difficult to distinguish walls from open space. This forces the drone to reduce its speed.


(e) t=94 s. The corridor widens enabling the drone to make a better map of its environment. The drone increases its speed.

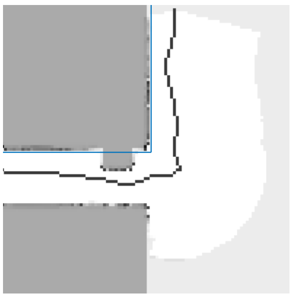
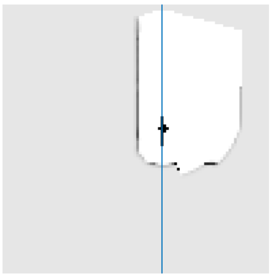
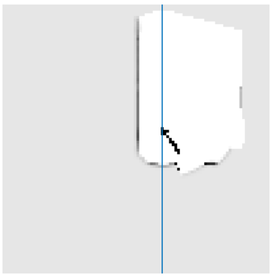

(f) The true obstacles superimposed on the occupancy grid map showing a good fit. There was an obstacle hidden around the corner that the drone managed to avoid by taking a larger turn.

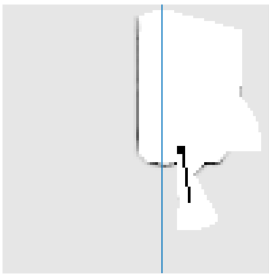
Figure 3.8: The drone is tasked to follow the blue lines downwards and to the left. The planned path is too close to the wall to fly safely. The drone is marked as a black dot and the planned path as the black line. The length of the line indicates the speed of the drone. The background color indicates the state of the occupancy grid, white is safe and darker colors indicate a higher probability of the cell containing an obstacle.



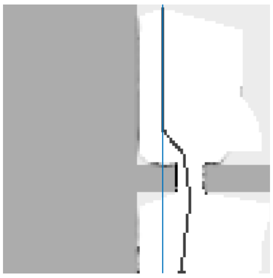
(a) The sensor cone size makes it difficult to detect if there is an opening in the wall. The drone reduces its speed and approaches slowly.



(b) The drone moves towards a possible opening.

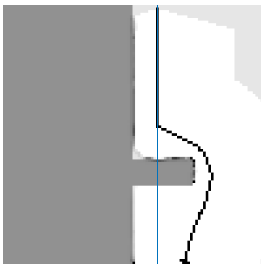


(c) An opening is found and the drone flies through at full speed.

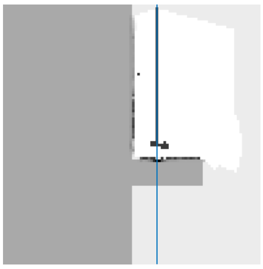


(d) The final path the drone followed. The true obstacles are superimposed on the occupancy grid map.

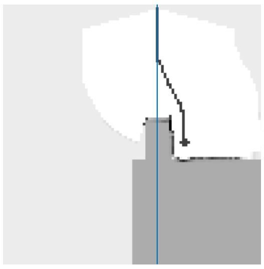
Figure 3.9: The drone is tasked to fly straight down, but the planned path does not take the small opening into account.



(a)



(b)



(c)

Figure 3.10: Figures 3.10(a) and 3.10(b) show that the drone manages to circumvent small obstacles, but fails at larger obstacles. Figure 3.10(a) is the largest obstacle the drone is able to circumvent with the current control actions and Δ. Figure 3.10(b) is slightly larger hindering the drone from circumventing it. Figure 3.10(c) shows that the drone can get stuck on the wrong side of an obstacle.

Occupancy grid maps attempt to merge dependent sensor information and independent information of multiple measurements in one map with one value at each cell which is bound to lose information. A problem caused by this simplification is that the cell probability value is dependent on the cell size. For small cell sizes, the probability does not act as predicted and the map converges to the a priori value much closer to the drone than the measured range. This is highlighted for the 1D case in [82]. As the problem is dependent on the number of possible map states it gets significantly worse with the number of dimensions making this strategy unfit for 3D. This also raises questions on how well the occupancy grid map manages to represent the true uncertainty, as the grid cell size should not affect the uncertainty in this manner.

A limitation of the sensor model used in this chapter is that it assumes a strict border between which cells that are within the field of view, and which are outside. In reality, a gradual transition is expected when using radars and sonars, where it becomes gradually less likely to detect obstacles the further to the side they are.

One of the goals of considering the probability of collision, rather than setting a fixed safety margin around the drone, was to use our knowledge on the uncertainty in different sensors and the motion of the drone when deciding how close to obstacles the drone should come. Explicitly considering the uncertainties could potentially cause the drone to act less conservatively, as a safety margin must be chosen conservatively large when not having a complete overview of all factors involved. But, as the probability values of the occupancy grid map cannot be interpreted literally, the resulting probability of collision does not have a literal meaning. This makes choosing the maximum acceptable probability of collision an ad-hoc process, similar to choosing a safety margin. A difference between directly using an ad-hoc safety margin and an ad-hoc probability of collision is that considering the probability enables the system to change its behavior when the uncertainty changes. This enables the drone to naturally consider changes in the uncertainty in estimated pose and speed and the dynamics of how the uncertainty develops in the future as shown in Section 3.5.2. Lastly considering uncertainty opens up for the next step of considering risk. For a crash-resistant drone, the speed can be used to quantify the potential consequences of contacting an obstacle. By considering risk the drone can potentially navigate areas with large uncertainty in the presence of obstacle if it moves at a low enough speed.

3.10 Future work

The main work that needs to be done within the mapping. A new method for mapping that generates more realistic probability values and is able to handle 3 dimensions should be developed. Additionally, a better sensor model that considers the gradual transition of the field of view is needed. Specular reflections and multipath are also important phenomena to consider. The collision avoidance strategy should also be tested out in 3D with a more realistic drone mode.

Chapter 4

Risk-based decision making

This chapter is based on the following publication

[47] **S. V. Rothmund**, C. A. Thieme, I. B. Utne, and T. A. Johansen, “A Bayesian Approach to Risk-Based Autonomy for a Robotic System Executing a Sequence of Independent Tasks,” *Submitted to Journal of Intelligent & Robotic Systems*, 2022. DOI: 10.36227/techrxiv.14054141.v3

The method was developed by S. V. Rothmund in collaboration with C. A. Thieme. Software development and simulations were done by S. V. Rothmund. Supervision was provided by I. B. Utne and T. A. Johansen. The first draft was written by S. V. Rothmund and revised by C. A. Thieme, I. B. Utne, and T. A. Johansen

4.1 Introduction

For a system to operate without direct human supervision, it must be able to evaluate the situation and handle deviations from normal operation [83]. These deviations are often connected with uncertainty, making it necessary to consider the risk of a task or operation. Risk can be defined as the “effect of uncertainty on objectives” [84]. Hagen et al. [6] argued that a system’s “ability to sense, interpret and act upon unforeseen changes in the environment and the [system] itself” is vital for achieving a high level of autonomy. Information on the state of real-world systems and environments is often uncertain or incomplete [85]. When acting with uncertain and incomplete information, the system cannot avoid making sub-optimal or erroneous decisions that it should detect and act to minimize the consequences of.

This chapter aims at developing a risk-based decision system that improves the ability of an autonomous system to interpret and act upon deviations from normal operation and to counteract the consequences of past erroneous or sub-optimal choices. This chapter focuses on operational decision-making for a robotic system executing a series of independent tasks, such as inspection or intervention at multiple locations.

Previous literature exists on making decisions based on uncertainty or risk. [37] uses a BBN to evaluate the collision risk during an under-ice operation with an autonomous underwater vehicle. Safety critical parameters, such as distance to the ice sheet, are automatically changed by considering how they affect the risk evaluated with the BBN. [38] chooses an emergency landing location for an unmanned aerial vehicle by evaluating the risk of the different landing locations with a BBN. [39] uses a BBN to evaluate the effect of different recovery and security strategies during a cyberattack against an industrial control system. Even though these works make risk-based decisions, they do not consider improving the system's ability to interpret its health state and the state of the environment.

An ability to infer the health state of the system based on indirect observations is demonstrated in [41], [86], [87] by using a DBN. This previous research does not consider how decision-making affects how the system develops, nor do they consider using the inferred health state for automatic decision-making.

Considering the action made by the system and using the inferred state of hidden variables for automatic decision-making has been done in educational systems [88]–[90] and dialog systems [91], [92]. These systems use a DDN to infer the state of the user based on their observed response to different actions made by the system. Even though these systems show some of the capabilities needed, they are made for a distinctly different type of problem, making them not directly applicable to automatic decision-making for robotic systems.

A system for inferring the state of the environment and the system based on indirect measurements, and using the results for automatic decision-making for a robotic system is presented in [40]. Here a DBN is used to infer the states of different underlying dynamic variables by combining information over time. The DBN uses this information to evaluate the probability that any of four pre-defined hazardous scenarios have occurred or are likely to occur in the future. Recovery actions are proposed if any of the scenarios are in a failed state, while preventive actions are proposed if any of the scenarios are in an anomalous state or expected to be in a failed state in the future.

A key difference between [40] and the work presented in this chapter, is that this chapter considers operational decision-making while [40] considers emergency fault handling. [40] therefore does not consider the risk and reward of executing different actions, how the choice of action affects how the system develops over time, nor does it consider inferring whether previous decisions were erroneous or sub-optimal.

The previously presented literature demonstrates that Bayesian models, such as BBN, DBN, and DDN, are promising tools for achieving this chapter's goal. Considering risk when making operational decisions was shown in [37], [38] to be a feasible approach. This further strengthens the case for Bayesian models as these model probabilistic relationships making them suitable to model risk [63].

This chapter develops a decision-making system for an autonomous robot executing a sequence of independent tasks. A DDN model is used which enables the system to combine direct measurements with the result of attempted task execution to infer the state of the system and the environment. A heuristic is proposed which evaluates if and how a task execution should be attempted, and whether a recovery action is needed. Additionally, the system is able to update its belief regarding past states thereby identifying previously attempted tasks that were wrongly skipped and should therefore be re-attempted.

To demonstrate the proposed method a case study of an industrial inspection multi-rotor drone is considered. The drone is tasked with mapping the thickness of metal surfaces in an industrial facility to identify damages to the structure. The measurements are conducted by contacting the surface with an ultrasound probe [93]–[97]. The large number of measurements needed to get sufficient coverage makes the operation costly for a human operator to directly and continuously monitor, thereby warranting the need for autonomous execution.

The contribution of this chapter compared to earlier literature is the development of an automatic risk-based system for operational decision-making, that is able to infer the state of the system itself and the environment based on indirect measurements, and that is able to evaluate past states with new information thereby enabling it to counteract past mistakes.

The rest of the chapter is structured as follows: Section 4.2 states the problem formulation. Section 4.3 presents the proposed method for developing and using the DDN. This method is applied to the case study in Section 4.4. Simulation results from the case study are presented in Section 4.5. Section 4.6 discusses the proposed method in light of the results from the case study. A conclusion is given in Section 4.7.

4.2 Problem statement

This chapter considers a robotic system that executes a sequence of independent tasks. Tasks are considered independent when "no task provides a necessary precondition for the fulfillment of another task" [98]. This chapter does not consider in which order the tasks should be executed. It is assumed that the tasks are given as an ordered list at the start of the operation. The tasks are assumed to be time-independent with no deadlines that have to be considered when planning.

This chapter considers a part of the autonomy layer that should decide if and how a task should be executed and whether a recovery action is needed. The task executions are assumed to have a probability of failing, in which case the system must decide if the task should be attempted again or skipped. Two classes of recovery actions are considered, requesting maintenance of the system, and returning to a previously failed task. This chapter assumes that a predefined set of possible execution actions are given for each task.

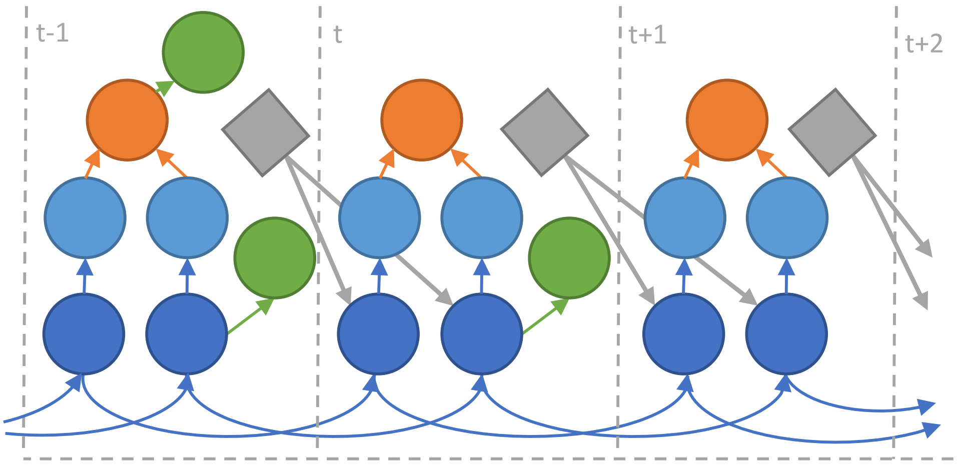


Figure 4.1: Simple DDN developed with the proposed method. Objectve nodes are shown in orange, failure cause nodes in light blue, condition nodes in dark blue, measurement nodes in green, and action nodes in gray. The current time step (t) is shown together with one earlier time step ($t-1$) and one future time step ($t+1$).

The available information before a task is attempted is insufficient to get a complete overview of the state of the drone and the environment. The robotic system must therefore incorporate information available on the outcome of previous actions to improve its situation awareness.

4.3 Method

This section presents the proposed method for developing the DDN that will be used to infer the state of the robotic system and the environment, together with a strategy for using the DDN to choose what action the robotic system should take.

Figure 4.1 gives an example of the resulting DDN for a simple problem with the proposed method. The basic procedure for using the DDN is as follows:

1. If available, insert evidence based on measurements available before a task is attempted.
2. Evaluate the risk and gain of executing different actions.
3. Execute the optimal action.
4. If available, insert evidence based on the observed outcome of the action.
5. Make a new time-step in the DDN. Each time step represents a decision that is made.

4.3.1 Developing the DDN

This chapter proposes developing the DDN through a top-down approach. This approach ensures that only states that can be distinguished from each other are included. The following steps are used to develop the DDN:

1. Describe the operation and system.
2. Model relevant objectives.
3. Model failure causes.
4. Model the condition of the failure causes.
5. Model dynamics.
6. Model measurements.
7. Quantify the CPTs.

Step 1 - Describe the operation and system

The operational description defines the tasks the system should execute and which actions the robotic system can choose between. This chapter considers three types of actions: executing the task, maintaining the system, and changing task.

Executing the task is associated with a probability of achieving the goal of the task and may lead to losses, which are discussed further in step 2. There can be different ways of executing the task with different costs associated with execution and with different consequences. There can be different ways of maintaining the system that repairs or mitigates different types of failures. Maintenance actions are associated with a direct cost that the system must weigh against the advantage of maintaining the system.

When changing task, the system can choose between going to the next task in the sequence or returning to a previous task. Leaving a task without fulfilling its goal is associated with a cost. This cost is weighed against the expected cost of attempting the task. How these trade-offs are made is discussed in detail in Section 4.3.2.

In the description of the robotic system, the available sensors and information from different subsystems, such as a navigation system, are given.

Step 2 - Model relevant objectives

As risk is the “effect of uncertainty on objectives” [84], the relevant objectives must be identified to make risk-based decisions. Two types of objectives are considered: achieving the task goal and avoiding hazardous events. A hazard is a set of adverse conditions that can lead to a loss [50]. Damage to the robotic system is one example of a loss. Relevant hazards can be identified through different risk analysis methods, such as preliminary hazard analysis (PHA) [99] or STPA [50]. A node is introduced in the DDN for every goal and hazard. Figure 4.1 shows a simplified case with only one goal or hazard node, shown in orange. These nodes take on a binary state indicating whether the objective will be met or not on this execution attempt.

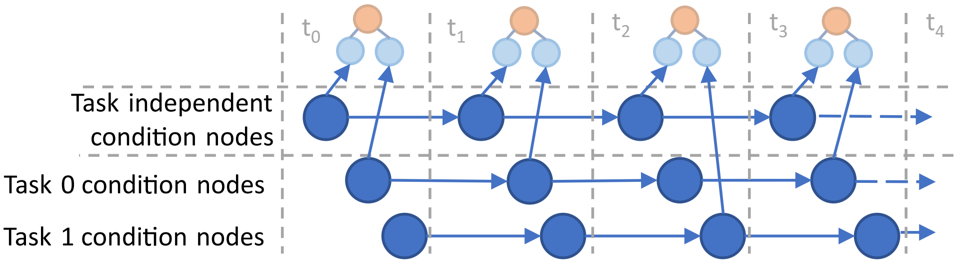


Figure 4.2: Example of how different task-specific nodes can be connected to the rest of the network at different time steps. Task 0 is connected to the rest of the network at time steps 0, 1, and 3, while task 1 is connected at time step 2.

Step 3 - Model failure causes

Different failure causes, such as faults in the robotic system and adverse environmental states, can prevent the objectives from being fulfilled. Not achieving an objective is considered a failure. The failure causes can be identified with a risk analysis; see [50], [99]. Nodes are introduced that represent groups of failure causes that cannot be distinguished from each other, shown as light blue in Figure 4.1. All failure causes that affect different measurements or that are affected differently by the choice of action can potentially be distinguished from each other. The failure cause nodes take on a binary state indicating whether any failure cause in this group will cause a failure on this execution attempt.

Step 4 - Model the condition of the failure causes

The failure cause nodes introduced in the last step consider the expected outcome of a single execution attempt. New nodes, called condition nodes, are introduced to model the general condition of the failure causes. These nodes could, for example, be defined as the amount of wear or the failure rate of a component. One condition node is introduced for each failure cause node as shown in dark blue in Figure 4.1. These nodes can have multiple states to model varying ability to achieve the goal.

Step 5 - Model dynamics

Using a dynamic model enables the system to incorporate information over time, thereby improving its estimates of the underlying conditions. A new time step is introduced in the DDN for each decision that is made. The condition nodes introduced in step 4 are connected to themselves between time-step as shown in Figure 4.1. Having a separate time step for each decision makes it possible to use the outcome of each action as a source of information.

Some conditions can be independent for each task. These conditions can be modeled by having an instance of the node for each task in the operation. The nodes representing the current task are connected to the current time step. An example of this is given in Figure 4.2.

Step 6 - Model measurements

Separate measurement nodes should be introduced to enable the modeling of measurement uncertainty. Measurements available before a task execution should depend on the condition nodes, while measurements of how the execution went should depend on the objective or failure cause nodes. Figure 4.1 shows how measurements, shown in green, can be available for condition nodes at the current time step and for condition nodes and on how the execution went at previous time steps.

Step 7 - Quantify the CPTs

The CPTs used by Bayesian networks can be quantified based on expert judgment and operational data. This enables the models to be used on novel systems where operational data is missing. Quantification of CPTs based on expert judgment is not a trivial task, and many different methods exist to simplify the process [63], [100]. This chapter simplifies the process by using Boolean operators to define which combination of failure causes that affect the different objective nodes. The CPT of the failure cause nodes that are children of condition nodes translates the condition into a probability of failure on this execution attempt. The CPTs of the condition nodes specify how the state can degrade or improve based on the choice of action. The CPTs of the measurement nodes quantify the measurement uncertainty.

4.3.2 Decision policy

Even though each task is independent, each decision that has to be made is not. This is due to the actions affecting the state of the drone, which will affect future actions. Finding the optimal decision policy when it's not sufficient to consider a single decision in isolation requires solving a POMDP, which in the general case is intractable except for small problems [68]. To circumvent this problem, a heuristic policy is proposed. The policy considers the following three strategies consisting of one or multiple actions: 1) move on to another task, 2) attempt to execute the task once and then move on to another task, or 3) execute a maintenance action, attempt to execute the task, and then move on to another task. The expected cost of each strategy is evaluated, and the first action of the cheapest strategy is executed. After executing the first action of the strategy, the optimal strategy is reevaluated. If strategy 2 is chosen multiple times in a row, then the system executes the current task multiple times without moving to another task. This ensures that the resulting closed-loop behavior can be closer to optimal behavior than any of the proposed strategies.

The cost of strategy 1, C_1, has only a cost if the goal of the current task is not achieved. This cost, C_G, is based on the consequence of not achieving the goal. This is shown in Equation (4.1). More cases can be added if there can be a partial fulfillment of the goal.

The cost of strategy 2, $C_2(e)$, depends on the choice of execution action, e. There is a direct cost for executing action e, $C_E(e)$, and an indirect cost if a hazardous event occurs. There can be multiple different hazards, each associated with its own cost, which are given as elements in the vector $\mathbf{C}_H(e)$. Experiencing a hazardous event can jeopardize the ability to successfully perform future task executions. This cost is not directly considered but should instead be compensated for when designing the cost of the hazard $\mathbf{C}_H(e)$. This cost can depend on the choice of execution action. If the execution does not achieve the goal of this task, then there will be the additional cost of moving to another task, C_1. The probability of achieving the task's goal, P_G, and the probability of different hazardous events occurring, $\mathbf{P}_H$, when executing an action are evaluated using the DDN. These values are found by evaluating the probability that the objective nodes are in a failure state at the current time step. The resulting cost function is shown in Equation (4.2). This cost is evaluated for all possible execution actions, e, applicable to the current task.

The cost of strategy 3, $C_3(m, e)$, depends on the choice of maintenance action, m, and execution action, e. The maintenance action can increase the probability of achieving the goal and reduce the probability of hazardous events occurring. The effect of the maintenance action is evaluated by inserting it as evidence in the action node at the current time step of the DDN and then simulating one step forward in time by temporarily adding a new time step to the DDN. The cost of execution (strategy 2) after performing maintenance m can then be evaluated at this time step, $C_{2,m}(e)$. The cost of the maintenance action must be included as well. This cost is often quite high but can improve the success rate of multiple future task execution attempts. The maintenance cost, $C_M(m)$, is divided by the expected number of executions until maintenance is needed again, $N(m)$. The resulting cost is shown in Equation (4.3) and should be evaluated for all combinations of maintenance actions, m, and execution actions, e.

$$C_1 = \begin{cases} 0 & \text{If the goal of the current task is achieved} \\ C_G & \text{Otherwise} \end{cases} \tag{4.1}$$

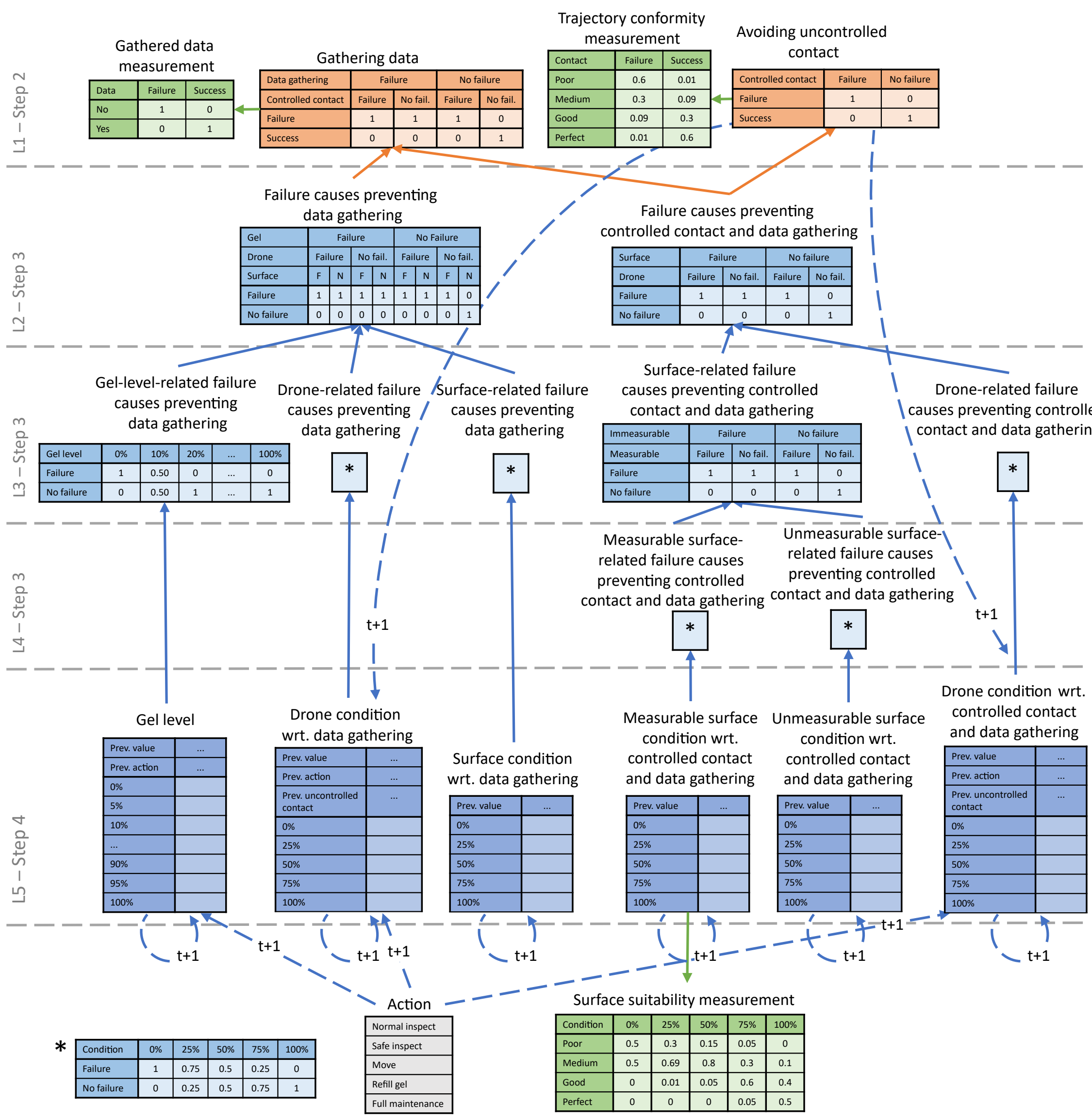
$$C_2(e) = C_E(e) + \mathbf{C}_H(e)^\top \mathbf{P}_H + (1 - P_G)C_1 \tag{4.2}$$

$$C_3(m, e) = C_M(m)/N(m) + C_{2,m}(e) \tag{4.3}$$

When moving to another task (strategy 1), the system can choose to revisit a previously attempted task. The expected cost of executing a previously attempted task is evaluated by simulating that the system moves to this task. The system returns to a previously attempted task if the expected cost of executing the task, $C_2(e)$, plus the cost of returning to the previous task, C_{Ret}, is lower than the cost of omitting the task, C_1, as shown in Equation (4.4). A task is reattempted if the visit is warranted for any of the available execution actions. If none of the previously attempted tasks are worth another attempt, then the system will move to the next task in the sequence that is not attempted.

$$C_{Ret} + C_2(e) < C_1 \tag{4.4}$$

Attempting a task before and after maintaining the system enables the system to


L1 – Step 2
L2 – Step 3
L3 – Step 3
L4 – Step 3
L5 – Step 4
Gathered data measurement
Gathering data
Trajectory conformity measurement
Avoiding uncontrolled contact
Failure causes preventing data gathering
Failure causes preventing controlled contact and data gathering
Gel-level-related failure causes preventing data gathering
Drone-related failure causes preventing data gathering
Surface-related failure causes preventing data gathering
Surface-related failure causes preventing controlled contact and data gathering
Drone-related failure causes preventing controlled contact and data gathering
Measurable surface-related failure causes preventing controlled contact and data gathering
Unmeasurable surface-related failure causes preventing controlled contact and data gathering
*
t+1
Gel level
Drone condition wrt. data gathering
Surface condition wrt. data gathering
Measurable surface condition wrt. controlled contact and data gathering
Unmeasurable surface condition wrt. controlled contact and data gathering
Drone condition wrt. controlled contact and data gathering
Action
Surface suitability measurement

Gathered data measurement

Data	Failure	Success
No	1	0
Yes	0	1

Gathering data

Data gathering	Failure		No failure	
Controlled contact	Failure	No fail.	Failure	No fail.
Failure	1	1	1	0
Success	0	0	0	1

Trajectory conformity measurement

Contact	Failure	Success
Poor	0.6	0.01
Medium	0.3	0.09
Good	0.09	0.3
Perfect	0.01	0.6

Avoiding uncontrolled contact

Controlled contact	Failure	No failure
Failure	1	0
Success	0	1

Failure causes preventing data gathering

Gel	Failure				No Failure			
Drone	Failure		No fail.		Failure		No fail.	
Surface	F	N	F	N	F	N	F	N
Failure	1	1	1	1	1	1	1	0
No failure	0	0	0	0	0	0	0	1

Failure causes preventing controlled contact and data gathering

Surface	Failure		No failure	
Drone	Failure	No fail.	Failure	No fail.
Failure	1	1	1	0
No failure	0	0	0	1

Gel-level-related failure causes preventing data gathering

Gel level	0%	10%	20%	...	100%
Failure	1	0.50	0	...	0
No failure	0	0.50	1	...	1

Surface-related failure causes preventing controlled contact and data gathering

Immeasurable	Failure		No failure	
Measurable	Failure	No fail.	Failure	No fail.
Failure	1	1	1	0
No failure	0	0	0	1

Gel level

Prev. value	...
Prev. action	...
0%	
5%	
10%	
...	
90%	
95%	
100%	

Drone condition wrt. data gathering

Prev. value	...
Prev. action	...
Prev. uncontrolled contact	...
0%	
25%	
50%	
75%	
100%	

Surface condition wrt. data gathering

Prev. value	...
0%	
25%	
50%	
75%	
100%	

Measurable surface condition wrt. controlled contact and data gathering

Prev. value	...
0%	
25%	
50%	
75%	
100%	

Unmeasurable surface condition wrt. controlled contact and data gathering

Prev. value	...
0%	
25%	
50%	
75%	
100%	

Drone condition wrt. controlled contact and data gathering

Prev. value	...
Prev. action	...
Prev. uncontrolled contact	...
0%	
25%	
50%	
75%	
100%	

*

Condition	0%	25%	50%	75%	100%
Failure	1	0.75	0.5	0.25	0
No failure	0	0.25	0.5	0.75	1

Action

Normal inspect
Safe inspect
Move
Refill gel
Full maintenance

Surface suitability measurement

Condition	0%	25%	50%	75%	100%
Poor	0.5	0.3	0.15	0.05	0
Medium	0.5	0.69	0.8	0.3	0.1
Good	0	0.01	0.05	0.6	0.4
Perfect	0	0	0	0.05	0.5

Figure 4.3: The conditional probability tables and dependencies in the DDN for the inspection drone case study. Some tables are intentionally left blank and are instead given in Tables 4.1 to 4.3. Nodes containing a * refer to the table marked with a * shown on the bottom left of the figure.

identify if a maintenance action helped. This behavior is encouraged by always choosing an execution action if the current task has not been attempted and if strategy 2 is cheaper than strategy 1. If this is not the case, then the normal policy is followed.

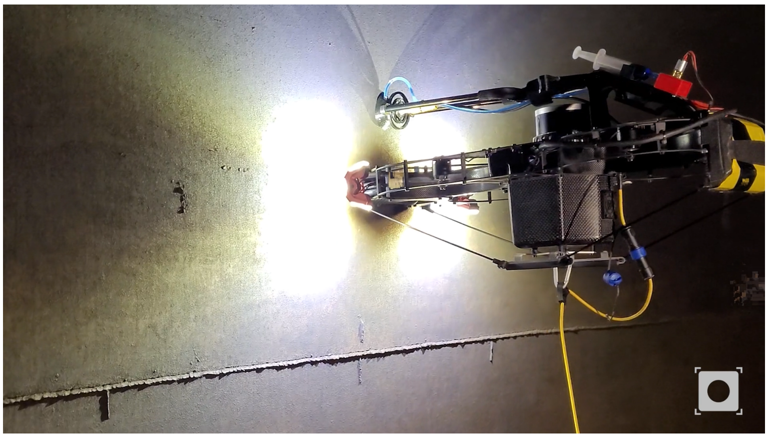


Figure 4.4: The ScoutDI drone during an ultrasound inspection of a storage tank. Courtesy ScoutDI.

4.4 Case study

In this section, the proposed method is applied to a multirotor drone tasked with industrial inspection. The case study setup is developed in cooperation with the drone inspection technology company ScoutDI. Figure 4.4 shows the ScoutDI drone performing an ultrasound thickness measurement. The case study is based on simulation.

4.4.1 Developing the DDN

Step 1 - Describe the operation and system

The operation consists of measuring metal surface thickness with an ultrasound sensor mounted on a multirotor drone. A large number of points are typically inspected. Every inspection point is considered a task in the proposed method. The system can choose between two different ways of inspecting the surface of the inspection point: a normal inspection and a slower but safer inspection. A small amount of gel is dispensed from a tank mounted on the drone for each inspection. One maintenance action available to the drone is to refill this tank. Another is to request a full maintenance check by an operator. The drone can skip inspection points deemed too costly to inspect autonomously.

The drone is equipped with a lidar used to detect obstacles and navigate.

Step 2 - Model relevant objectives

The goal of each task is to measure the surface thickness of the inspection points. The drone is assumed to operate in controlled industrial facilities consisting of metal surfaces without any humans present. This makes damage to the drone the most relevant loss. A hazard that can cause this loss is uncontrolled contact with a surface or other object. Nodes representing the two objectives are shown on line L1 in Figure 4.3.

Step 3 - Model failure causes

Through discussions with ScoutDI different failure causes were identified. Some of the failure causes, such as an empty gel tank, rust or dirt stuck on the ultrasound sensor, or inspection surfaces covered with rust or dirt can prevent data from being gathered. Other failure causes, such as a worn motor, poor navigation quality, or obstacles, can lead to uncontrolled contact in addition to preventing data from being gathered. To simplify modeling, two intermediate nodes are introduced: one for failure causes preventing data from being gathered, the other for failure causes preventing both controlled contact and data from being gathered. These are shown on line L2 in Figure 4.3.

The drone and the surface of the inspection point are affected differently by choice of action. Executing an inspection may damage the drone, while the surface will not be affected. Similarly, maintaining the drone does not affect the surface. Moving to a new inspection point will change the surface but not affect the drone. A distinction between drone-related and surface-related nodes is therefore needed. Furthermore, the refill-gel action only affects the gel level. These nodes are shown on line L3 in Figure 4.3.

Before an inspection is executed, a lidar scan of the inspection surface can reveal protruding obstacles that will prevent controlled contact and data gathering. The limited resolution of the lidar can cause it to systematically miss thin obstacles, such as welding joints or minor surface irregularities. A distinction between failure causes that are measurable and those that are not can therefore be made, as shown on line L4 in Figure 4.3.

Step 4 - Model the condition of the failure causes

A slightly dirty or uneven surface, or a minor fault in the drone, can reduce the likelihood of an inspection succeeding without hindering it completely. For all nodes except the “gel level” node, the states of the condition nodes reflect the average frequency at which the respective conditions will cause a failure. These frequencies are discretized into different states, as shown on line L5 in Figure 4.3.

The state of the gel level indicates the amount of gel left. When the gel level approaches zero, an insufficient amount of gel might be deployed. This will prevent data from being gathered.

Step 5 - Model dynamics

Drone-related conditions have a probability of degrading with each inspection attempt. The probability and severity of the degradation depend on whether an uncontrolled contact occurred and whether a normal or safe inspection was performed. The gel level is gradually depleted with each inspection attempt. There can

Table 4.1: Initial probability distributions.

	Gel level	Drone condition wrt. data gathering	Surface condition wrt. data gathering	Unmeasurable surface condition wrt. controlled contact and data gathering	Measurable surface condition wrt. controlled contact and data gathering	Drone condition wrt. controlled contact and data gathering
0%	0	0.03	0.3	0.02	0.01	0.005
25%	0	0.03	0.05	0.005	0.01	0.005
50%	0	0.04	0.05	0.005	0.01	0.005
75%	0	0.5	0.2	0.005	0.07	0.01
100%	1	0.4	0.4	0.965	0.9	0.975

Table 4.2: Probability of transitioning to worse states given the choice of action and whether an uncontrolled contact is avoided. The table gives the probability of degrading the state by different amounts. Transitions that lead to negative probabilities are omitted before the resulting distribution is normalized. The probability of transitioning to a better state is 0.

Node name	Drone condition wrt. data gathering				Drone condition wrt. controlled contact and data gathering			
Action	Normal inspect		Safe inspect		Normal inspect		Safe inspect	
Avoiding uncontrolled contact	Failure	Success	Failure	Success	Failure	Success	Failure	Success
-100%	0.025	0.005	0.005	0.0025	0.025	0	0.005	0
-75%	0.025	0.005	0.005	0.0025	0.025	0	0.005	0
-50%	0.025	0.005	0.005	0.0025	0.025	0	0.005	0
-25%	0.025	0.005	0.005	0.0025	0.025	0	0.005	0
-0%	0.9	0.98	0.98	0.99	0.9	1	0.98	1

be some variation in the amount of gel dispensed making the number of inspection attempts before a refilled uncertain.

The surface-related conditions are assumed constant over time and independent at each inspection point. These are handled as discussed in Section 4.3.1 step 5 and illustrated in Figure 4.2.

Step 6- Model measurements

The "surface suitability measurement" is introduced as shown at the bottom of Figure 4.3. This measurement is, as discussed in step 3, based on how flat the area around the inspection point seems based on the lidar scan.

After an inspection is executed, a measurement of how the execution went is needed. Whether data is successfully gathered is readily available from the ultrasound thickness sensor. Whether an uncontrolled contact occurred cannot be directly measured. Instead, this can be inferred based on the trajectory conformity measurement. This measurement is made by comparing the observed trajectory of the drone with the intended trajectory and identifying any deviations in position, velocity, and heading.

Table 4.3: Probability of reducing the gel level by different amounts.

	Gel level
-15%	0
-10%	0.1
-5%	0.8
-0%	0.1

Step 7 - Quantify the CPTs

Contact-based inspection drones are in the early stage of development. It is, therefore, little operational data and experience to use as a basis for the quantification. The choice of hardware and software design will significantly affect the quantification process. The quantification will be sensitive to factors such as how robust the ultrasound sensor is, how robust the drone is to impact, and how well the drone manages to navigate. To demonstrate the proposed algorithm, some example values are chosen in collaboration with ScoutDI. The following assumptions were considered during the quantification process:

- Some inspections require the operators to clean the inspection surface first [101]. As this is not possible for the drone, there is a chance that there will be surfaces where the drone cannot gather data.
- The drone must be in stable contact to get a measurement. Touching the wall correctly with the sensor is difficult, making it likely that the drone will fail at some inspection attempts.
- The sensor can become defect due to dirt or rust sticking to it. This can happen even without an uncontrolled contact occurring
- An uncontrolled contact can displace the sensor or damage the drone's integrity, making it unable to continue. The likelihood of damaging the drone is low as it is built to be robust to impacts.

The result of the quantification process is shown in Figure 4.3 and Tables 4.1 to 4.3. The initial probability distributions can be found in Table 4.1. Tables 4.2 and 4.3 show the probabilities of transitioning to a worse state for the different drone condition nodes when an inspection is attempted. The refill gel action will set the gel level to 100%. The full maintenance action sets all drone-related nodes, including the gel level, to their initial distribution.

4.4.2 Decision policy

The decision policy presented in Section 4.3.2 is used with the parameters given in Table 4.4. These costs are based on the expected time use of the different actions. The expected time use of an uncontrolled impact is based on the expected time needed to repair the different degrees of damages that can occur times the likelihood of them occurring from an uncontrolled impact. It is assumed that an uncontrolled contact will seldom damage the drone, making the cost relatively low. Even if the

Table 4.4: The different costs used in the decision policy for the case study.

Symbol	Cost
$C_E(e = \text{Normal inspect})$	0.5 min
$C_E(e = \text{Safe inspect})$	1 min
$C_H(e = \text{Normal inspect})$	20 min
$C_H(e = \text{Safe inspect})$	10 min
$C_M(m = \text{Refill})$	10 min
$C_M(m = \text{Full maintenance})$	60 min
$N(m = \text{Refill})$	20
$N(m = \text{Full maintenance})$	30
C_G	10 min
C_{ret}	0.5 min

drone is not damaged, it might require human assistance if it falls to the ground. The cost of not achieving the goal is based on the additional time used for a manual inspection. Evaluating an exact value for these costs can be difficult in practice. Some tuning of the values might therefore be necessary if the observed behavior is inadequate. Having values that can be interpreted still gives an advantage as it gives an intuition on what the values should be.

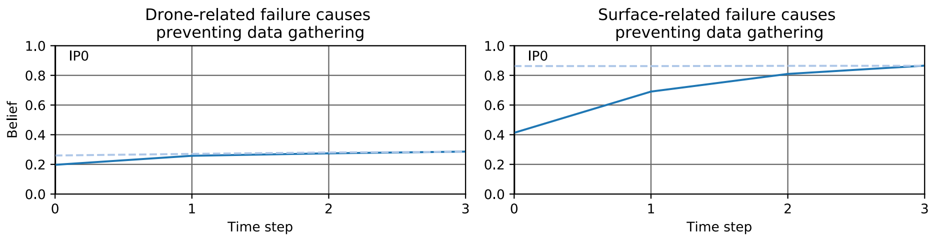
4.5 Results

This section presents four scenarios for the inspection drone case study. These scenarios are chosen to demonstrate how the proposed approach fulfills the five previously defined requirements. The scenarios represent different types of failures and events that are deemed likely to occur during the drone's mission. Scenario 1 considers a case where the ultrasound sensor is not working. In Scenario 2 the drone is unable to have a controlled contact. Scenario 3 demonstrates the effect measurements have on the system's behavior. Lastly, Scenario 4 considers a case where the gel is depleted.

The simulations are done by having a model of the drones state and the sate of the different inspection surfaces as state machines. Their respective states define which values that are measured that are made and what the output will be of different actions. The drone's state can either be working, with a defective ultrasound sensor, or in a state making it unable to make controlled contact. Each inspection surface can either be ideal, unable to be measured, with a measurable blocking obstacle, or with an immeasurable blocking obstacle. The DDN used to make decisions is evaluated using the SMILE [60] library for Python.

Scenario 1 - Sensor defect

In this scenario, the ultrasound sensor is not working. All inspections end with no data being gathered but perfect trajectory conformity. Figures 4.5 to 4.7 show how the belief of the system develops over time when new inspections are attempted.



Time step	Measured failure causes	Chosen action	Measured objective fulfillment
0	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
1	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
2	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
3	Surface suitability: Perfect	Move to IP1	

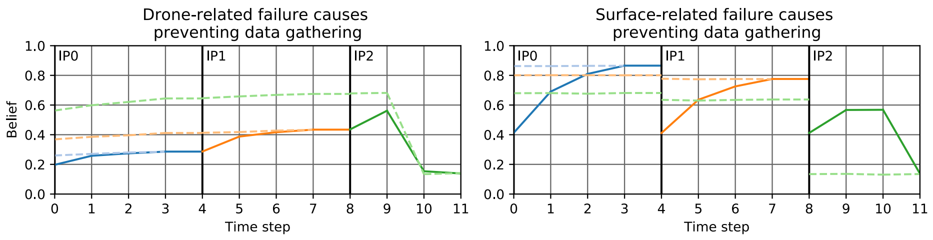
Figure 4.5: Scenario 1. The table shows the measurements available before inspection, the choice of action, and the resulting measurements. The graph shows the state of failure causes relevant in this scenario. The solid line shows the belief of the drone that a failure cause is present at each time step. The dashed line shows the updated belief of past states evaluated every time the drone moves to a new inspection point (IP), which is marked with a vertical line. The color of the dashed line indicates when the updated belief was evaluated.

Only the belief that drone-related and surface-related failure causes will prevent data gathering are shown. The rest of the failure causes have a belief close to 0 throughout this scenario.

Figure 4.5 shows the behavior and beliefs of the drone when it is at the first inspection point. As seen in the table in Figure 4.5, the drone attempts to execute an inspection, which results in no data but perfect trajectory conformity. After the first inspection fails, the belief that surface-related failure causes prevent data from being gathered increases, as shown by the solid blue line. The belief that drone-related failure causes prevent data gathering also increases but much less. This is due to it being more probable that a single failed inspection is caused by the surface than by the drone. This trend continues for the subsequent inspection attempts. At time step 3, the belief that surface-related failure causes will prevent data gathering is high enough, making the drone skip the current inspection point and move on to inspection point 1.

The dashed blue line in Figure 4.5 shows the system's belief about past states evaluated at time step 3. As the stat of the surface cannot change, the belief about the past states is equal to the newest belief. The state of the drone can, on the other hand, degrade, making the updated belief regarding the state of the drone at time step 0 slightly lower than at time step 3.

Figure 4.6 is a continuation of Figure 4.5 that includes the beliefs, measurements, and actions at inspection points 1 and 2 as well. At inspection point 1, the same behavior is observed as at inspection point 0. After three failed attempts, the system skips this inspection point and moves on to inspection point 2. When evaluating



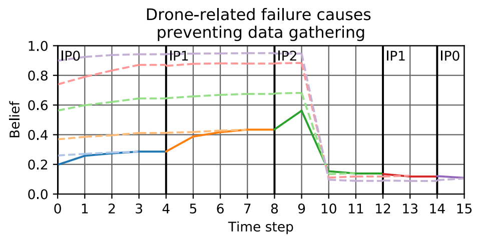
Time step	Measured failure causes	Chosen action	Measured objective fulfillment
4	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
5	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
6	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
7	Surface suitability: Perfect	Move to IP2	
8	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
9	Surface suitability: Perfect	Full maintenance	
10	Surface suitability: Perfect	Normal inspect	Data: Yes, Trajectory conformity: Perfect
11	Surface suitability: Perfect	Return to IP1	

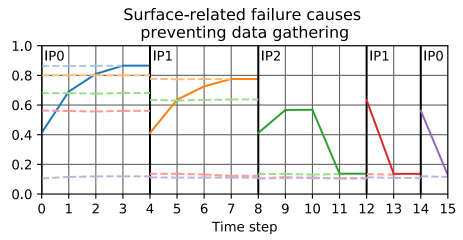
Figure 4.6: Scenario 1 part 2, continuation of Figure 4.5. The table shows the measurements available before inspection, the choice of action, and the resulting measurements. The graph shows the state of failure causes relevant in this scenario. The solid line shows the belief of the drone that a failure cause is present at each time step. The dashed line shows the updated belief of past states evaluated every time the drone moves to a new inspection point (IP), which is marked with a vertical line. The color of the dashed line indicates when the updated belief was evaluated.

the past states at time step 7, shown with the orange dashed line in Figure 4.6, the probability that the drone-related failure causes are preventing data gathering has increased. Since the belief that drone-related failure causes prevented data gathering in time steps 0-3 has increased, the belief that surface-related failures caused the failed inspection at inspection point 0 decreases. This can be seen by the dashed orange line being lower than the dashed blue line at time step 0-3 for the surface-related failure causes.

After failing an inspection at inspection point 2 as well, the belief that the drone-related failure causes prevent data gathering is high enough, making a full maintenance worth the cost. After maintenance, the following inspection at time step 10 is successful. As the inspection failed before the maintenance but succeeded after, it becomes more probable that there was a fault with the drone that was solved by the maintenance. Reasoning backward in time decreases the probability that surface-related failures caused the previously failed inspections, as shown with the dashed green line in Figure 4.6

When considering where to go next, the system evaluates whether a previously visited inspection point is worth another inspection attempt. Since the belief that the surfaces on these inspection points caused the failures has decreased, the system concludes that they are worth another attempt. Figure 4.7 shows how the system first visits inspection point 1 again, where data is gathered successfully. This further strengthens the belief that the drone caused the previously failed inspections. The





Time step	Measured failure causes	Chosen action	Measured objective fulfillment
12	Surface suitability: Perfect	Normal inspect	Data: Yes, Trajectory conformity: Perfect
13	Surface suitability: Perfect	Return to IP0	
14	Surface suitability: Perfect	Normal inspect	Data: Yes, Trajectory conformity: Perfect
15	Surface suitability: Perfect	Move to IP3	

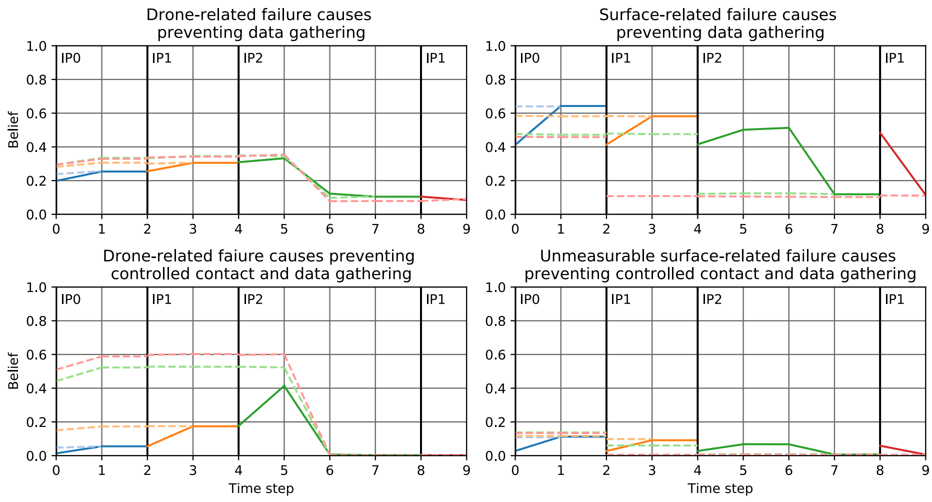
Figure 4.7: Scenario 1 part 3, continuation of Figures 4.5 and 4.6. The table shows the measurements available before inspection, the choice of action, and the resulting measurements. The graph shows the state of failure causes relevant in this scenario. The solid line shows the belief of the drone that a failure cause is present at each time step. The dashed line shows the updated belief of past states evaluated every time the drone moves to a new inspection point (IP), which is marked with a vertical line. The color of the dashed line indicates when the updated belief was evaluated.

system then returns to inspection point 0 and has a successful inspection before moving on to a new inspection point.

Scenario 2 - Inability to get controlled contact

The drone is in a condition such that it cannot establish good contact with the surface. All inspections result in data not being gathered and medium path conformity. The belief that drone-related and surface-related failure causes prevent data gathering and that drone-related and unmeasurable surface-related failure causes prevent controlled contact and data gathering is shown in Figure 4.8. The rest of the failure causes have a belief close to zero throughout this scenario.

In this scenario, the drone does not attempt to inspect the inspection point again after the failed inspection attempt at time step 0, as shown in Figure 4.8. This is due to the large cost associated with the possibility of having uncontrolled contact if the inspection is reattempted. At inspection point 1, a safe inspection action is performed since there is a considerable probability that the failure at time step 0 was caused by the drone. The system attempts one last inspection at inspection point 2 before requesting full maintenance. After maintenance, a safe inspection is executed since the failure might have been caused by the surface, which was unaffected by the maintenance action. As the inspection was successful, the belief that surface-related failure causes prevented controlled contact and data gathering at inspection point 2 decreased. When reasoning backward in time at time step 7, as shown by the dashed green line, the belief that “drone-related failure causes prevents controlled contact and data gathering” at time steps 0 and 2 is significantly increased. This decreases the belief that the failed inspection was caused by surface-



Time step	Measured failure causes	Chosen action	Measured objective fulfillment
0	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Medium
1	Surface suitability: Perfect	Move to IP1	
2	Surface suitability: Perfect	Safe inspect	Data: No, Trajectory conformity: Medium
3	Surface suitability: Perfect	Move to IP2	
4	Surface suitability: Perfect	Safe inspect	Data: No, Trajectory conformity: Medium
5	Surface suitability: Perfect	Full maintenance	
6	Surface suitability: Perfect	Safe inspect	Data: Yes, Trajectory conformity: Perfect
7	Surface suitability: Perfect	Return to IP1	
8	Surface suitability: Perfect	Safe inspect	Data: Yes, Trajectory conformity: Perfect
9	Surface suitability: Perfect	Move to IP0	

Figure 4.8: Scenario 2. The table shows the measurements available before inspection, the choice of action, and the resulting measurements. The graph shows the state of failure causes relevant in this scenario. The solid line shows the belief of the drone that a failure cause is present at each time step. The dashed line shows the updated belief of past states evaluated every time the drone moves to a new inspection point (IP), which is marked with a vertical line. The color of the dashed line indicates when the updated belief was evaluated.

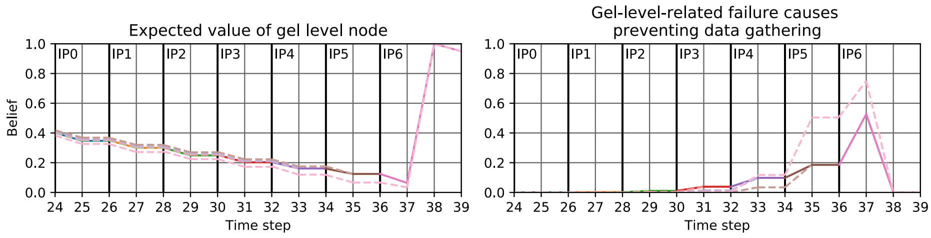
related failure causes, making another attempt worth its cost. A safe inspection is performed at inspection point 1, as there could still be surface-related failure causes at this inspection point.

Scenario 3 - Surface suitability

This scenario demonstrates how the surface suitability measurement affects the choice of actions. Figure 4.9 shows how the system decides not to attempt an inspection if the surface suitability measurement is poor. With a medium surface suitability measurement, a safe inspection is attempted, but the system only attempts one inspection. When the surface suitability measurement is good but not perfect, two inspection executions are attempted before moving on.

Time step	Measured failure causes	Chosen action	Measured objective fulfillment
0	Surface suitability: Poor	Move to IP1	
1	Surface suitability: Medium	Safe inspect	Data: No, Trajectory conformity: Perfect
2	Surface suitability: Medium	Move to IP2	
3	Surface suitability: Good	Safe inspect	Data: No, Trajectory conformity: Perfect
4	Surface suitability: Good	Normal inspect	Data: No, Trajectory conformity: Perfect
5	Surface suitability: Good	Move to IP3	
6	Surface suitability: Perfect	Normal inspect	Data: Yes, Trajectory conformity: Perfect
7	Surface suitability: Perfect	Move to IP4	

Figure 4.9: Scenario 3. The table shows the measurements available before inspection, the choice of action, and the resulting measurements. No graphs are shown as they give little additional information in this scenario.



Time step	Measured failure causes	Chosen action	Measured objective fulfillment
...	...	...	...
33	Surface suitability: Perfect	Move to IP5	
34	Surface suitability: Perfect	Normal inspect	Data: Yes, Trajectory conformity: Perfect
35	Surface suitability: Perfect	Move to IP6	
36	Surface suitability: Perfect	Normal inspect	Data: No, Trajectory conformity: Perfect
37	Surface suitability: Perfect	Refill gel	
38	Surface suitability: Perfect	Normal inspect	Data: Yes, Trajectory conformity: Perfect

Figure 4.10: Scenario 4. The table shows the measurements available before inspection, the choice of action, and the resulting measurements. All inspections prior to time-step 33 resulted in data being gathered and perfect trajectory conformity. The graph shows the state of failure causes relevant in this scenario. The solid line shows the belief of the drone that a failure cause is present at each time step. The dashed line shows the updated belief of past states evaluated every time the drone moves to a new inspection point (IP), which is marked with a vertical line. The color of the dashed line indicates when the updated belief was evaluated.

Scenario 4 - Gel level

This scenario demonstrates the effects of the gel level node. Figure 4.10 shows the expected value of the gel-level node in addition to the belief that gel-level-related failure causes prevent data gathering to better show how the gel depletes over time. The figure starts after 12 successful inspections. With each inspection, the expected gel level decreases. The belief that the "gel-level-related failure causes preventing data gathering" first increases when the expected gel level is close to depleted. When no data is gathered in the inspection attempt at time step 37, the drone assumes a low gel level caused it, making it execute a refill.

4.6 Discussion

Scenario 1 shows that the system is able to distinguish between faults with the drone and adverse inspection surfaces by combining information over time. This enables the system to executing maintenance actions when needed. Furthermore, this scenario demonstrates that reasoning backward in time enables the system to realize that the previously visited inspection points were not the cause of the failure as it previously assumed. This enables the system to return to previously failed tasks and reattempt the inspection.

Scenario 2 shows a case similar to scenario 1, with the difference that the drone experienced a worse trajectory conformity. This could be explained by an unmeasured obstacle in this current location, by a damage to the drone, or it could be a random failure. The possibility that there was an unmeasurable obstacle made the system not reattempt the failed task, as it did in scenario 1, but rather go directly to the next task. The possibility that there was a failure with the drone made the drone execute a safe inspection at the second location. This demonstrates how the system reasons with risk, and how it considers the underlying causes while doing so.

Scenario 3 demonstrates that the system considers the measurements available before the task execution to proactively manage risk. Scenario 4 demonstrates how the gel-level node affects the behavior. In scenario 1, the system does not believe that the gel level caused the failed inspections, as the failure occurs immediately after take-off. In scenario 4, many inspections were successfully performed before the execution failed, making it probable that the gel was depleted. This shows that the system manages to distinguish between different types of internal faults when it affects the system differently.

The proposed method for building the DDN ensures that the condition nodes, which are the possible explanations for the observed behavior, are quite general. Having general nodes ensures that the nodes actually represent features the system is able to distinguish based on the observations. When there is a high belief that "drone-related failure causes prevent data gathering", the system does not know what the failure cause is. It could be anything preventing the drone from gathering data at multiple inspection points that do not affect its motion. The sensor could be displaced, there might be dirt on the sensor, or the sensor might be wrongly calibrated or unsuitable for the current mission. Which of these scenarios is true is irrelevant, as they all prevent data from being gathered and have the same solution: requesting maintenance. Constructing general condition nodes for all possible ways the system can be affected by actions and measurements ensures that the system has a possible explanation for all observations.

The DBN produced by the proposed method does not model the severity of the losses that can be caused by the occurrence of a hazard, such as having an uncontrolled contact. The different losses that can occur may have different severity and probabilities associated with them. Modeling the losses could enable the system to distinguish between different levels of severity, enabling the system to change its behavior accordingly.

Based on the observed results, no obvious sub-optimal behavior with the proposed heuristic decision policy was observed. One drawback with the heuristic is the discount factor, $N(m)$. This factor could, in theory, be based on the probability of degrading the drone with each inspection attempt. This factor has a straightforward interpretation and effect on the resulting behavior, making the discount factor an acceptable trade-off between simplicity and quality of the heuristic.

The resulting decision policy needs to evaluate the network multiple times for each time step to simulate the effect of different maintenance actions and to evaluate whether the system should return to a previous point. This could potentially be alleviated by further simplifications, such as specifying thresholds for the different condition nodes. A predefined maintenance action can then be executed when the belief surpasses the threshold. The drawback of this approach is that it would lose information on the interaction between components. This is especially important for more complicated systems with more causal factors.

The point of the case study was to demonstrate capabilities that can be achieved with the proposed system. It was not to solve the case study in the most optimal or simplest manner. Similar behavior as the presented results could be achieved by, for example, defining an exhaustive set of conditional rules on which action to perform for each possible set of measurements that are made. An example of such a rule would be to always skip a task if it has failed three times, and always repair if three tasks are skipped. These types of methods may work for very simple problems but do not scale well for more complex problems as the possible combination of measurements that require special rules often grows very quickly. By instead estimating what the hidden state of the system and environment is and by modeling the effect of actions, as done in this chapter, this problem is alleviated.

Evaluating DDNs becomes computationally expensive when the number of time steps increases. The number of time steps can be limited by using a sliding window approach where only the n newest time steps are included in the DDN [56]. The initial condition of the DDN must reflect the information that is no longer inside the sliding window. This can be achieved by setting the priors at the first time step inside the window equal to the posterior evaluated at the last step outside the window. A drawback with a sliding window approach is that only time steps inside the window will be considered when evaluating past states with new information. This constitutes a challenge for the proposed method as the number of time steps that are computationally feasible to consider might be too low to consider all previously attempted tasks that are interesting to reconsider. One possible way to alleviate this problem is to find a more compact way to represent information on previous tasks. Currently, previous tasks are represented by multiple time steps, one for each execution attempt and a time step for every move and repair action.

4.7 Conclusion

This chapter presents an approach for structuring a DDN and using it for operational decision-making. The chapter's goal is to contribute toward enabling autonomous systems to safely operate without direct human supervision. Through a case study of an industrial inspection drone it is demonstrated how the resulting system is able to increase the drones situation awareness about its own state and the state of the environment, and how the drone can use this information to make risk-based decisions. Additionally it is demonstrated how evaluating past states with new information can reveal tasks the drone wrongly skipped that it should return to for another inspection attempt.

Future work can consider how the severity of a hazard occurring can be modeled, how evaluating past states could be simplified such that a longer time horizon can be considered, and on experimental validation.

Acknowledgment

We would like to thank ScoutDI for cooperation on the case study. Specifically we would like to thank Morten Fyhn Amundsen for discussion regarding the quantification and Kristian Klausen for general discussions on how the system works and potential failure causes.

Chapter 5

Supervisory risk control

This chapter is based on the following publication

[49] **S. V. Rothmund**, C. A. Thieme, I. B. Utne, and T. A. Johansen, "Supervisory Risk Control with Application to Industrial Drone Inspection," *Submitted to Autonomous Robots*, 2022. DOI: 10.36227/techrxiv.21287334

The method was developed by S. V. Rothmund and C. A. Thieme. Software development and experimentation were done by S. V. Rothmund. Supervision was provided by I. B. Utne and T. A. Johansen. The first draft was written by S. V. Rothmund and C. A. Thieme, revisions were provided by I. B. Utne, and T. A. Johansen

5.1 Introduction

To safely increase the level of autonomy of robotic systems operating outside of controlled environments it is essential to enable the system to identify changes in the environment and the robotic system itself and to adapt to the identified changes [83]. To explore this topic, this chapter considers a case study of a tethered industrial inspection drone. The drone, shown in Figure 5.1, is designed for indoor visual inspection of hard-to-reach places in industrial facilities. This drone is manually controlled but should be simple and safe to operate even for unskilled operators. The goal of this work is to give the drone the necessary situation awareness and adaptability to enable unskilled operators to safely operate the drone. The definition of situation awareness given in [12] is adopted in this work. This definition considers the system's ability to sense and comprehend the current state and project how the state will develop in the future. This chapter focuses on the comprehension part of situation awareness.

It is proposed in [35] that giving the robotic system a holistic understanding of risk can give it the situation awareness needed to ensure safety. Several research articles have used a risk measure to control robotic systems. Most of the articles consider

Figure 5.1: The ScoutDI tethered drone used as the case study. Courtesy ScoutDI.

a single, or a few, sources of risk (often called hazards), such as the variance in the current and future position of the robotic system and dynamic obstacles [20]–[23], variance in the future position of the robotic system and mapping uncertainty [24], [46], uncertainty in traffic density [102], or time, distance, and/or relative speed to obstacles [14]–[17]. These works demonstrate how risk information can increase the situation awareness of the system, but do not produce the holistic understanding needed to ensure safety.

The possibility of including multiple risk sources which could give a holistic understanding of risk is discussed in [30]. They use a risk map when planning the path of an autonomous ship. This risk map could be built based on multiple factors such as sea-state, visibility, and maneuverability. The concept of using risk maps to aggregate different sources of risk has been explored in [31]–[34] as well. [44], [45] presents a concept called Safety-Driven Behaviour Management. This concept consists of a situation model that is used to evaluate the risk level. [44], [45] present different general approaches for extracting information from the environment, building the situation model, and assessing the risk. However, they do not consider how decisions are to be made based on the risk level.

Another concept is presented in [35] called supervisory risk control. They propose to use a risk model that is built based on the results of risk analysis in the control of autonomous systems. A risk analysis is a systematic process to identify what can go wrong, how likely it is, and what the consequences would be [64]. Performing a risk analysis gives a foundation for which factors that should be included in the model and how they relate to each other. In [35] the results of a STPA [50] are used as a basis to make a BBN [103] that can evaluate the risk during operation. [35] does not consider how the BBN is to be used for online risk control. Different possibilities for using the result of the risk analysis in control systems are discussed in [104]. Different approaches where BBNs are used for decision-making can be found in [38]–[41].

Both [36] and [37] continue the work in [35] by developing supervisory risk controllers for different case studies with the proposed method. [36] controls the machinery mode of an autonomous ship by considering, amongst others, the risk evaluated with a BBN during operation. Their work uses an STPA as the basis for making the BBN, as in [35], but extends the method to also consider consequences. [37] uses the same general idea but uses a hazard identification (HAZID) as the basis for making the BBN instead. The resulting supervisory risk controller decides on an adequate distance to the ice sheet for under-ice autonomous underwater vehicle operations.

Several articles have presented risk analyses for different drone operations [105]–[113]. None of these works have considered using the results of the risk analysis for online decision-making. Different BBN models of drone operations can be found in [42], [43], [105], [106], [114]–[117]. Only [42], [43] consider using the models online to monitor the system, but they do not consider using the model for automatic decision-making. Among the surveyed literature, only [44], discusses using an online risk model for automatic decision-making for drone operations, but considers it outside of their scope. None of the surveyed literature are using a holistic risk model for automatic decision-making to ensure safe drone operations.

The goal of the supervisory risk controller is to reduce the risk of accidents by modifying safety-critical parameters during operation. This can be compared to the goal of run-time assurance which is to ensure safety. Run-time assurance, as presented in [118], considers an operational envelope, fixed limits on different states of the system, that ensures that no accident will occur if the limits are upheld. By instead using risk, as in the supervisory risk controller, the system can get a holistic overview where it can consider how the different states interact with each other to create hazardous events. A fixed separate threshold on each state will most often be a more conservative approach, as not considering the interconnections must be compensated for by increasing the safety margins. Real-time assurance, as presented in [118], ensures safety by switching from a non-assured complex function to a simpler but assured function when the system approaches the limits of the operational envelope. The assured function shall then guarantee that the system never leaves the limits of the operational envelope, and preferably move the system back to the state where the complex function can again be used. This scheme ensures that the whole system stays within an operational envelope even though it employs a complex function that cannot be assured. The supervisory risk controller takes a different perspective by instead employing a risk model made on the basis of a risk analysis to reduce the risk of accidents. Run time assurance and a supervisory risk controller have different domains where they are important, and many systems could potentially benefit from having both.

The contribution of this chapter is to further develop the method proposed in [35], and to apply and experimentally test it on an industrial inspection drone case study. This work extends the methods proposed in [35]–[37] by modeling the relationship between causal factors and the available measurements, and by extending the BBN used in these works to a DDN. These changes enable the system to identify the

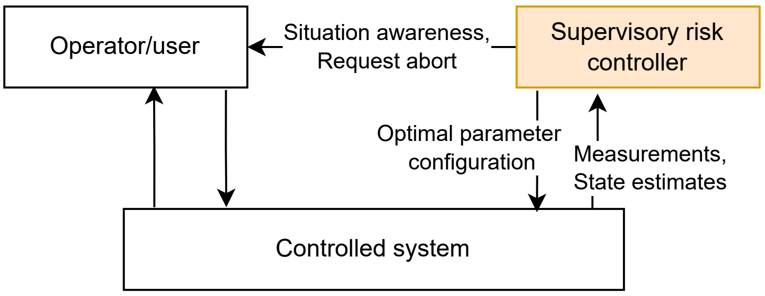


Figure 5.2: Overview of a supervisory risk controller that monitors a system and modifies parameters that determine how the system acts.

state of causal factors by combining information from different measurements over time, thereby increasing the situation awareness of the system. This work is the first to apply a real-time supervisory risk controller in an experimental setting where it changes parameters affecting the behavior of a robotic system to ensure a safe operation.

The rest of the chapter is structured as follows. Section 5.2 outlines the general method which is applied to the inspection drone case study in Section 5.3. Results from the experiments are presented in Section 5.4. The results and method are discussed in Section 5.5 before a conclusion is given in Section 5.6.

5.2 Method

The work in this chapter considers a supervisory risk controller [35] as shown in Figure 5.2. The supervisory risk controller sits on top of the existing control system and monitors how the system operates. It uses this information, together with information supplied by the human operator, to build its situation awareness. The supervisory risk controller can affect the operation in three different ways. One of the ways is to modify parameters in the different lower-level controllers to ensure an acceptable risk of continuing the operation. The second is to request that the human operator aborts the mission if no parameter choice makes the risk acceptable. Lastly, the supervisory risk controller forwards its information about the situation to the human operator to increase their situation awareness.

In [35], a two-phase process for designing the risk model that will be used in the supervisory risk controller is proposed. The chapter adds a third phase that is considering the control algorithm itself:

- **A:** Perform risk **analysis** to identify how the system can fail.
- **M:** Develop a risk **model** to evaluate the risk during operation based on the risk analysis.
- **C:** Develop a **control** algorithm that based on the risk model changes control

system parameters identified in the risk analysis and evaluates whether an abort should be requested.

The risk analysis is performed to identify causal factors and analyze how they can lead to a loss. Performing a risk analysis increases the chance that potentially disastrous events are considered and handled. The purpose of the risk model is to evaluate the risk during operation based on the factors identified with the risk analysis. Having a single model that combines all factors enables the system to get a holistic understanding of the situation.

The supervisory risk controller, as shown in Figure 5.3, uses the risk model, which is continuously updated with new measurements, to evaluate the optimal parameter configuration. Modifying control system parameters based on the risk level during operation enables the robotic system to adapt to the current situation and make risk-based decisions. In the opposite case, with a static parameter choice, a trade-off has to be made between safety in a worst-case situation and efficiency under normal conditions. Adapting the control parameters based on the risk allows the system to use conservative parameters in worst-case conditions and liberal parameters under normal conditions, thereby increasing both safety and efficiency.

An overview of the process of making the supervisory risk controller is given in Figure 5.4, which is explained in more detail in the subsequent sections. When following this procedure it can become necessary to revise previous steps with the knowledge gathered from subsequent steps.

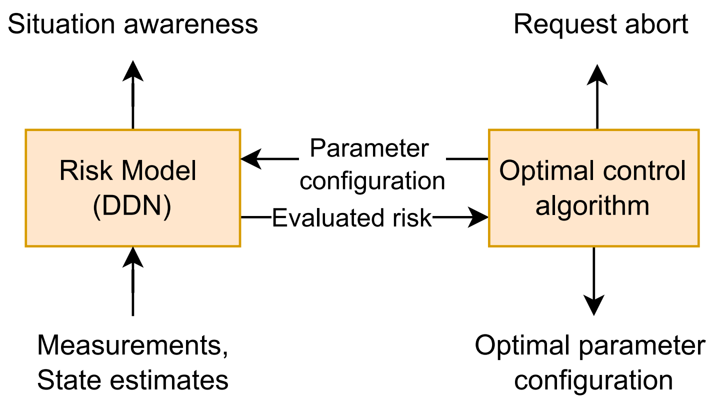

Situation awareness
Request abort
Risk Model (DDN)
Parameter configuration
Evaluated risk
Optimal control algorithm
Measurements, State estimates
Optimal parameter configuration

Figure 5.3: The internal workings of the supervisory risk controller shown in Figure 5.2. A risk model is gradually updated with measurements and state estimates. An optimal control algorithm finds the optimal parameter configuration by evaluating the risk of different configurations using the risk model.

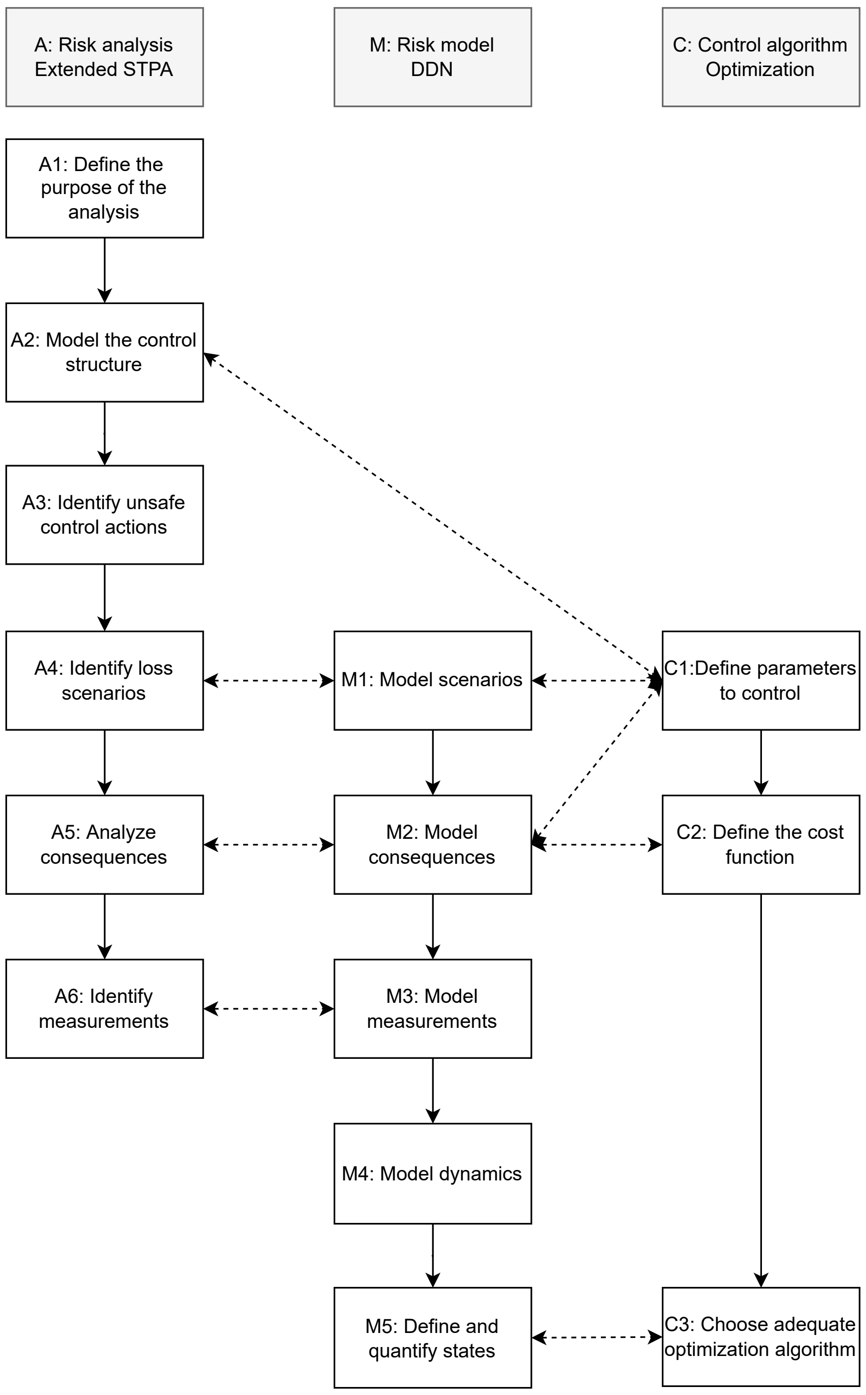

A: Risk analysis
Extended STPA
M: Risk model
DDN
C: Control algorithm
Optimization
A1: Define the purpose of the analysis
A2: Model the control structure
A3: Identify unsafe control actions
A4: Identify loss scenarios
A5: Analyze consequences
A6: Identify measurements
M1: Model scenarios
M2: Model consequences
M3: Model measurements
M4: Model dynamics
M5: Define and quantify states
C1:Define parameters to control
C2: Define the cost function
C3: Choose adequate optimization algorithm

Figure 5.4: Overview of the proposed process when using an extended STPA for risk analysis, a DDN for risk modeling, and an optimization algorithm for control. Dashed lines represent interactions between the three phases.

5.2.1 Risk analysis

The first part of the supervisory risk controller design is to perform a risk analysis. This analysis should be done by risk specialists together with domain experts who have experience operating the system under investigation. When possible, historical data from operations, or data from controlled experiments, should be used to give insight into the system under investigation.

[35] proposes to use STPA based on [50] for the first phase of the analysis (A). Although there are other relevant methods, the advantage of STPA is that it has a systems approach that enables it to identify emergent failures which occur due to the interactions of system components [50]. This property makes STPA suitable for analyzing robotic systems due to their complex interactions of software and hardware components. STPA has been shown to identify accident scenarios that are difficult to identify with other methods [119].

The original purpose of the STPA was to identify constraints for the system design. As this chapter considers using the results for automatic decision-making two additional steps are added. Steps A1 to A4 are based on [50], while step A5 is similar to the expansion proposed in [36], and step A6 is a novel contribution in this chapter.

A1: Define the purpose of the analysis by defining the system that is to be analyzed, together with the system boundaries, which losses should be considered, and system-level hazards that can cause these losses.

A2: Model the system as a hierarchical control structure. This model describes the system as a set of controllers that interact with control actions and feedback. Each controller has a process model that defines what it knows about the rest of the system and the environment. The responsibilities of each controller are defined.

A3: Identify unsafe control actions (UCAs). These are actions that in some context can cause a hazard. This step should not consider whether it's possible or likely for such an action to be performed in this context.

A4: Identify scenarios that describe how an UCA can occur.

A5: Identify how different causal factors affect the severity of losses caused by the scenarios identified in step A4.

A6: Identify measurements that give information on causal factors. Measurements in this context are not limited to direct sensor readings but can be any information available to the supervisory risk controller during a mission.

5.2.2 Risk model

The second phase of the supervisory risk control design process is to develop the online risk model. The risk model evaluates the risk, probability distribution over consequences, of continuing the operation based on the information available to the model.

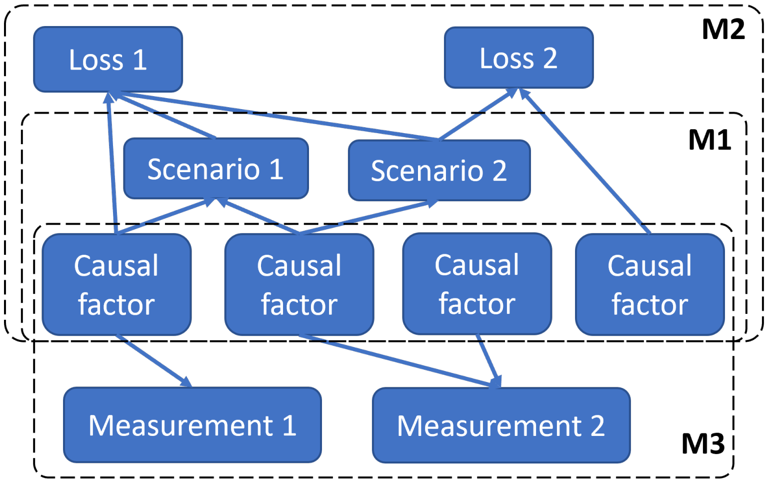


Figure 5.5: Example structure for the resulting DDN. M1 through M3 refer to steps introduced in Section 5.2.2.

Following [35] a BBN is constructed based on the result of the STPA. A BBN is used since it can handle many different types of factors where precise relationships are not known, and as it naturally evaluates probabilities. These features enable BBNs to produce the holistic risk awareness needed. This chapter extends the BBN to a DDN allowing the model to incorporate information over time and to consider the decisions made by the system.

The following steps are based on [35] but are expanded here to include the information gathered in steps A5-A6 and to consider a dynamic model.

M1: Model the relationships between causal factors and the occurrence of the scenarios defined in step A4.

M2: Model the relationship between scenario occurrence and losses based on step A5.

M3: Model the relationship between causal factors and measurements based on step A6.

M4: Model dynamics by connecting causal factors across time.

M5: Define the states of the nodes in the network and quantify the relationships between nodes based on historical data, data from controlled experiments, and expert judgments.

An example of how the resulting network can look is given in Figure 5.5.

5.2.3 Supervisory risk control algorithm

The last phase of the supervisory risk controller design is the development of the control algorithm itself. Different approaches for utilizing risk measures in control algorithms are presented in [104]. In this work, a cost optimization approach is chosen as it can directly use the evaluated risk in its cost function. The general process for using the risk model to find the optimal set of control parameters is as follows:

1. Insert observations on the measurement nodes in a new time step in the DDN.
2. Find the choice of parameters that minimizes a cost function that considers the risk. The risk is evaluated by first inserting the parameter choice as evidence in the parameter nodes in the new time step of the DDN and then by evaluating the probability distribution of the loss nodes at this time step.
3. Apply the optimal parameter choice to the relevant controllers.

The control algorithm is designed with the following steps:

C1: Chose which parameters to modify.

C2: Define the cost function considering both the risk and the cost of the parameter choice.

C3: Choose an optimization algorithm.

The parameters that are relevant are factors identified in steps A4 and A5 that can be modified by the supervisory risk controller during operation.

The cost function should consider the different risks, probability distribution over consequences, for the different losses, and the reduction in operational efficiency or system capabilities due to changing the parameters. How the parameter choice affects the operation must be quantified and included as a term in the cost function. The risks should be constrained to ensure an acceptable risk level. If no choice of parameters fulfills the constraints then the parameters that minimize the risk should be chosen and a mission abortion should be requested.

The resulting optimization problem is in the general case a nonlinear integer program without derivative information. This is due to the DDN being able to model nonlinear interactions, having discrete states, and not having derivative information available. If no more assumptions can be made, then the optimal solution can only be found by searching over all possible combinations of parameter choices. If the DDN and the cost function are constructed such that the local optimum of the cost function is equal to the global optimum then algorithms such as the hill climbing algorithm [48] can be used. If this is not the case then heuristic algorithms such as simulated annealing [48], [120] can be used to get a good but not necessarily optimal solution.

5.3 Industrial drone inspection

A case study of an industrial inspection quadcopter drone developed by the drone inspection technology company ScoutDI is considered. The drone is used to visually inspect hard-to-reach locations in industrial environments. A human operator controls the velocity of the drone using a joystick. An obstacle avoidance system modifies the velocity command based on information from a lidar mounted on the drone to prevent collision with obstacles. The drone is connected to a ground station with a tether which ensures reliable communication and supplies power to the drone. Note that an experimental development drone was used in this work. Crucial parts of the drone, such as the obstacle avoidance algorithm and parts of the hardware configuration, differ from the ones ScoutDI use in their present customer-ready products. Any risks identified in this chapter, therefore, do not necessarily represent risks in the customer-ready products of ScoutDI.

5.3.1 Risk analysis

This risk analysis was done in cooperation with ScoutDI through multiple workshops, informal meetings, and stress testing of the system. The initial workshop was performed with the lead software engineer in the spring of 2021 and followed the STPA procedure [50] as presented in Section 5.2.1. This was followed up by informal meetings with the lead hardware engineer and CTO through the summer and early fall of 2021. These meetings were more of a loose brainstorming process, where different UCAs from the STPA were used as starting points. The results from these meetings were used to update the initial STPA. The stress testing was done in the spring of 2022 and consisted of exposing the drone to situations that were identified in the workshop and in the informal meetings to be potentially hazardous. These tests gave insight into how the drone works, how it can fail, and how different failure causes can be measured. Getting information from the lead software and hardware engineer together with the CTO ensured a wide coverage of different potential problems.

A1 - Define the purpose of the analysis

The goal of this case study is to develop a supervisory risk controller that monitors the drone's operation and modifies different control parameters to ensure an acceptable risk level. Focus is therefore placed on onboard components of the drone that are related to obstacle avoidance, navigation, and control. The operator is located outside the room where the drone operates, there are therefore no humans present in the drones operating environment. The analysis will therefore focus on the following losses:

L1: Damage to drone or facility.

L2: Mission aborted.

The hazards that can lead to these losses (losses marked with square brackets), which the supervisory risk controller has some control over, are as follows:

H1: Physical contact with obstacle [L1][L2].

H2: Loss of controlled flight [L1][L2].

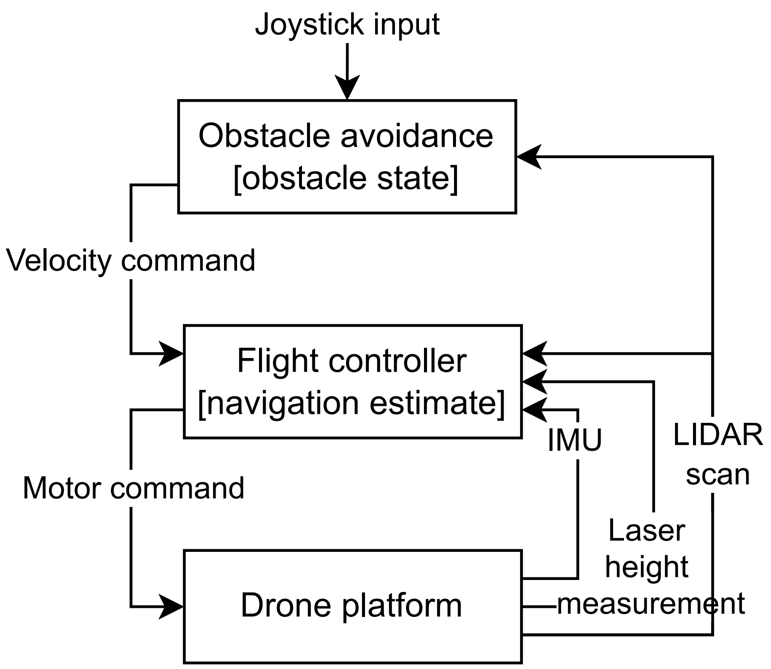


Figure 5.6: Control structure for the inspection drone case study. The process models needed by the different controllers are marked in brackets.

A2 - Model the control structure

The control structure is shown in Figure 5.6, which consists of an obstacle avoidance module, the flight controller, and the drone platform.

The obstacle avoidance system gets a joystick input from the human operator and generates a velocity command which it sends to the flight controller. The obstacle avoidance system is responsible for ensuring that the drone follows the safe part of the joystick input while maintaining a larger distance than the safety distance to all obstacles. This case study applies a simple obstacle avoidance algorithm that removes the part of the joystick input that points toward the closest obstacle closer than the safety distance. Additionally, a velocity is added that pushes the drone away from the closest obstacle closer than the safety distance. The obstacle avoidance system needs a process model of the location of obstacles relative to the drone. This process model is derived from lidar measurements. As there can be dust or other particles in the air that may cause sporadic reflections a dust filter is applied. This dust filter removes points close to the drone that do not have a sufficiently close point in the previous lidar scan.

The flight controller evaluates motor commands for each of the four motors. The primary responsibility of the flight controller is to keep the drone stable, the secondary responsibility is to follow the velocity command given by the obstacle avoidance module. This means that the flight controller will ignore the velocity command when needed to stabilize the drone. To control the drone, the flight controller has a process model of the navigational state of the drone. An extended Kalman filter (KF) is used to integrate information from the lidar and the inertial measurement unit (IMU).

The drone platform consists among other things of rotors, sensors, and a power controller. The power controller supplies the motors with the correct voltage to reach the motor command provided by the flight controller. The drone platform is in this analysis considered the controlled process.

The responsibilities of the controllers are as follows:

R1: The obstacle avoidance system is responsible for keeping the drone further away from obstacles than the safety distance [H1].

R2: The obstacle avoidance system is responsible for following the safe part of the joystick input.

R3: The flight controller is responsible for generating a motor command that keeps the drone stable and in the air [H2].

R4: The flight controller is responsible for generating motor commands that follow the velocity commands close enough to ensure that the drone does not deviate outside of the safety distance [H1].

Hazards relevant to the different responsibilities are marked in square brackets.

A3 - Identify unsafe control actions

The STPA handbook [50] provides four categories for UCAs. These are how the control action can cause a hazard by: 1) not being provided, 2) being provided, 3) being provided too late or out of order, 4) being stopped too soon or applied too long. These categories are useful when the controllers communicate with discrete or sporadic commands, such as activating a module. Since this case study only considers controllers that continuously send each other time-varying control inputs, these categories were of less help.

Instead, it was considered how the controllers could fail to fulfill their responsibilities. Based on this approach the following UCAs were identified:

U1: Obstacle avoidance system provides a velocity command towards an obstacle closer than the safety distance [H1].

U2: Obstacle avoidance system does not provide a velocity command away from an obstacle that is closer than the safety distance [H1].

U3: Flight controller provides motor commands that are inadequate to keep the drone within the safety distance of the velocity command [H1].

U4: Flight controller provides motor commands that are inadequate to stabilize the drone [H2].

Relevant hazards are marked in square brackets.

A4 - Identify loss scenarios

When defining the scenarios it is useful to have a list of possible scenario categories. The STPA handbook [50] gives a long list of categories that should be considered. Many of these categories, however, are most relevant during the design of the system. The following refined points based on [50] were considered the most interesting for supervisory risk control:

- How can measurement errors make the process model wrong thereby causing an UCA?
- How can the process model be wrong even with correct measurements thereby causing an UCA?
- How can environmental conditions cause a control action to become unsafe?
- How can the control algorithm generate UCAs even with a correct process model and sufficient environmental conditions?
- How can commands from a higher level controller cause an UCA?
- How can the actuator that is executing the control command or communication channel transmitting the command cause it to become unsafe?

The different loss scenarios were identified by going through the list above for each of the UCAs. The resulting full list of scenarios is found in Section 5.7. As the goal of this chapter is to work as a proof of concept, it is considered outside the scope to consider all of the identified scenarios. A subset of the scenarios that are considered is given in Table 5.1. Note that all scenarios related to computational hardware and navigational state estimation were chosen to be outside the scope of this chapter.

Table 5.1 also contains causal factors that were identified as substantially affecting the likelihood of the scenario occurring. When evaluating causal factors it is necessary to decide on an adequate level of detail. The approach used in this chapter was to start by introducing quite general causal factors and then refining them when parts of the factors were relevant for other scenarios, losses, or measurements.

While stress testing the drone it was found that even when the drone was standing still with no disturbance, good motors, low tether tension, and a large distance to walls, floor, and ceiling, the motor load was still varying quite a lot. The motor load could suddenly increase for a few seconds before going down again to the expected level. This phenomenon will be referred to as *varying motor effects* in the rest of the chapter as its cause was not identified.

Table 5.1: Refined scenarios based on the complete list given in Section 5.7. Parameters that the supervisory risk controller can control are marked with a **[P]**.

ID	Scenario	Causal factors
	Scenarios related to [U1] and [U2]	
S1:	Thin obstacles that are detected in only some of the lidar scans are filtered away as noise causing the drone to come closer to the obstacle than the safety distance.	Environment unobservability
	Scenarios related to [U3]	
S2:	Safety distance smaller than the braking distance of the drone causes the drone to violate the safety distance.	Maximum speed **[P]** Safety distance **[P]**
S3:	External disturbance causing violation of the safety distance.	Disturbance Safety distance **[P]**
S4:	Loss of control authority due to motor saturation causes violation of the safety distance.	Motor wear Motor load to counteract tether tension Maximum vertical acceleration **[P]** Motor load to counteract disturbances Varying motor effects Safety distance **[P]**
	Scenarios related to [U4]	
S5:	Physical contact with obstacle causes loss of controlled flight.	Physical contact with obstacles [H1] Speed **[P]**
S6:	Motor failure causes loss of controlled flight.	Motor wear
S7:	Excessive disturbance causes loss of controlled flight.	Disturbances

Table 5.2: Measurements providing information on the identified causal factors.

Measurement	Causal factors
Number of points filtered away by the dust filter	Environment unobservability Dust in the air
Amount of dust on the camera feed	Dust in the air
Roll/pitch error (measured compared to reference)	Disturbances
Motor load	Motor wear Motor load to counteract tether tension Motor load to counteract disturbance Varying motor effects
Drone tilt (static roll/pitch angles)	Diagonal tether tension
Laser down distance	Height over ground

A5 - Analyze consequences

The potential consequence to the drone if it contacts an obstacle is that it can induce a large enough disturbance to cause loss of controlled flight, as described in scenario [S5]. Increasing the speed of the drone increases the kinetic energy which makes this more likely. The drone can also contact an obstacle in a way that causes a loss of control even at lower speeds. This is more likely to occur when the drone loses control authority due to motor saturation as the drone then moves in an uncontrolled manner.

Loss of control will in all cases cause the mission to be aborted [L2] as the drone has to be checked for damages before the mission can be continued. Additionally, it can cause damage to the drone or the facility [L1]. The severity of the damages depends mainly on the potential energy of the drone as the drone operates at relatively low speeds. This means that the damage potential increases with the height over the ground.

A6 - Identify measurements

Information on how different causal factors can be measured is based on the stress testing of the system. The stress test exposed the system to situations where the causal factors given in Table 5.1 were in an adverse state. It was then analyzed how this affected different measurable quantities. It was also analyzed which other factors affected the measurements as the measurement could also be caused by factors that do not increase the risk. The resulting list of measurements is given in Table 5.2.

Hard-to-detect obstacles, such as wire fences and welding joints, may only sporadically be detected by the lidar scans. These sporadic measurements may be removed by the dust filter. The number of points filtered away by the dust filter, therefore, does not only give information on the amount of dust in the air but also on the environment unobservability. Whether there is dust in the air can be seen with the camera. In this proof-of-concept, the human operator must evaluate the presence of airborne dust from the camera feed and forward it to the supervisory risk controller.

The level of external disturbances affecting the drone can be estimated by comparing the roll and pitch angle setpoints given by the flight controller with the observed roll and pitch angles. The observed roll and pitch angles should be a delayed version of the commanded value due to the dynamics of the drone. Any roll or pitch angles that are not caused by a command are caused by disturbances. To compensate for delays, the commanded values for the last 1 second are recorded, and the maximum and minimum values are extracted. If the measured value is between the maximum and minimum then the error is set to 0. If not, then the error is equal to how much larger or smaller the value is than the maximum or minimum. The Euclidean norm of the roll and pitch angle errors are taken to produce a single value. Note that constant offsets such as diagonal tether tension will not greatly affect this measurement.

Increases in the load of the motors are in addition to joystick inputs caused by having to counteract tether tension, disturbance, motor wear, and quickly varying motor effects. When evaluating the motor load, the expected effect of the joystick input should be simulated and removed such that the measurement only gives information on the state of the environment and the drone. Experiments showed that the increased motor load by accelerating in the horizontal plane was minimal, only when changing the height of the drone was there a noticeable effect. To avoid having to evaluate the expected motor load from the joystick, focus was placed on horizontal motion during most of the experiments.

The motor load needed to lift the tether can be directly calculated when the tether hangs straight down. The tether may hang at an angle if it is approaching full extension, it is stuck, or it is exposed to a lot of friction. In these cases, the tether tension may exceed what is expected when considering height alone. If the tether is at an angle, then a horizontal force will act on the drone. The drone must then tilt to counteract this force, which can be measured by the roll and pitch angle measurements. How much motor load is needed to counteract the increased tether tension caused by the diagonal tether depends on factors that are not measured. The additional load must be zero when the drone is not tilted and is more likely to be large with larger drone tilts. The tilt of the drone is evaluated by low-pass filtering the roll and pitch angles with a time constant of 3 s to filter away quick variations due to disturbance and changes in joystick input. The tilt is then the Euclidean norm of the low-passed roll and pitch angles.

5.3.2 Risk model

Figure 5.7 shows the DDN made from the results of the risk analysis by following the steps outlined in Section 5.2.2.

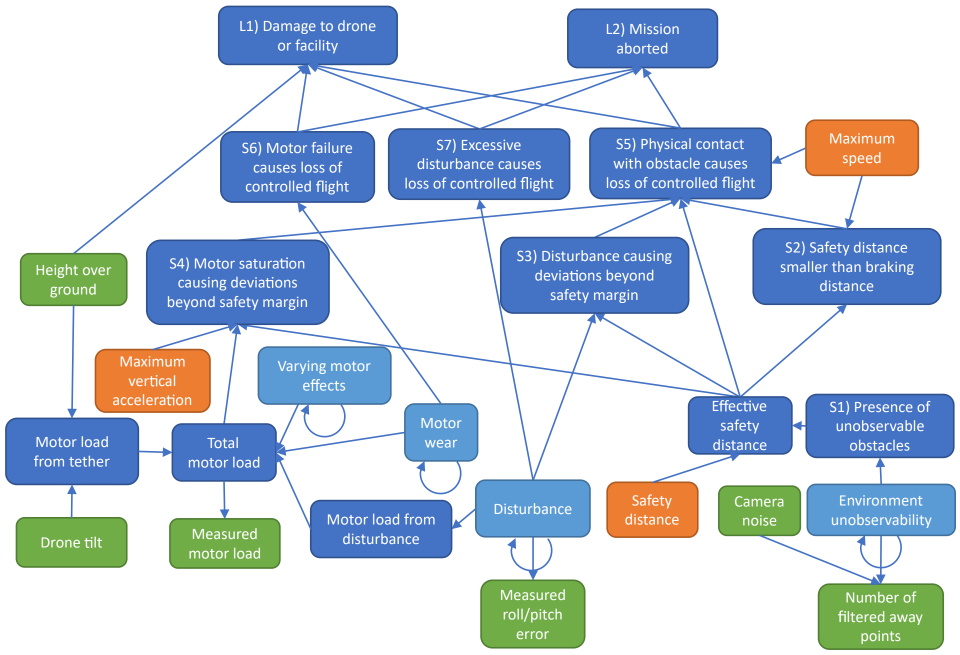

L1) Damage to drone or facility
L2) Mission aborted
S6) Motor failure causes loss of controlled flight
S7) Excessive disturbance causes loss of controlled flight
S5) Physical contact with obstacle causes loss of controlled flight
Maximum speed
Height over ground
S4) Motor saturation causing deviations beyond safety margin
S3) Disturbance causing deviations beyond safety margin
S2) Safety distance smaller than braking distance
Maximum vertical acceleration
Varying motor effects
Motor wear
Effective safety distance
S1) Presence of unobservable obstacles
Motor load from tether
Total motor load
Disturbance
Safety distance
Camera noise
Environment unobservability
Drone tilt
Measured motor load
Motor load from disturbance
Measured roll/pitch error
Number of filtered away points

Figure 5.7: DDN for the industrial inspection drone. Dark blue nodes represent intermediate, scenario, and loss nodes. Parameters the supervisory risk controller can control are shown in orange, measurement nodes in green, and underlying causal factors in light blue. The causal factors are dynamic. [L1] and [L2] represent losses introduced in Section 5.3.1 step A1. [S1] to [S7] represents scenarios introduced in Section 5.3.1 step A4

M1 and M2 - Model causal factors, scenarios, and losses

The scenarios related to inadequate obstacle avoidance ([S2], [S3], [S4]) do not directly cause a loss but can do so through scenario [S5]. Scenario [S1] does not directly cause a loss but negates the effect of the *safety distance* which can make scenarios [S2], [S3], [S4], and [S5] more likely.

M3 - Model measurements

Some of the measurements were introduced as separate nodes that are affected by the underlying causal factors. Introducing separate measurement nodes enables the modeling of measurement uncertainty. A separate node was not introduced when the uncertainty was deemed insignificant to the resulting behavior.

M4 - Model dynamics

The causal factors that are not directly observable (such as *height over ground*) or are known parameters (such as *safety distance*) are made dynamic by connecting them to their previous state. An update frequency of 1s was chosen for the network as it was considered sufficiently fast to react to changes in the drone and its environment.

M5 - Define and quantify states

The CPTs used by BBNs, DBNs, and DDNs can be quantified based on operational data and expert judgments. Even though the stress testing gave some data, it was far from sufficient for direct quantification. Experience gathered from the stress testing was used as a basis for quantifying based on an expert judgment.

There are multiple challenges when quantifying CPTs based on expert judgments. One type of challenge relates to psychological biases that affect the judgment process [121]. Another type comes from determining all the values of the CPTs. The size of the CPTs quickly grows large when the number of parent nodes and the number of discrete states increases. Determining all the values in a consistent manner can be challenging. Different approaches for alleviating this challenge are presented in [122]. All of the presented methods define some anchor points, add some uncertainty, and then use different interpolation functions to fill in the rest of the values.

A different approach is followed in this work. First, a qualitative description of how we expect the nodes to relate to each other is given. A mathematical function is then constructed to behave similarly to the qualitative description. This step is similar to the different interpolation methods surveyed by [122], but with the difference that knowledge about how the nodes should relate to each other is used in the design of the function. The next step is to tune parameters in the function by setting different input values and testing if the output is as expected. The parameters of the function are then modified until they give reasonable output values. This step is similar to specifying anchor points. We found this process easier to follow as it more directly could utilize our knowledge of input-output-relationships.

The list of descriptions, equations, and the definition are given in Tables 5.3 to 5.6, where $\mathcal{N}(\mu, \sigma)$ represents a normal distribution with mean μ and standard deviation σ, $\mathcal{B}(p)$ represents a Bernoulli distribution that takes the value 1 with probability p, $\mathcal{U}(a, b)$ represents a uniform distribution between a and b, and $\mathcal{E}(a)$ represents an exponential distribution with parameter a. The states of all nodes are limited to either be in the Boolean set $\{0, 1\}$, or to be in the real-valued interval $[a - b]$. Values outside the interval are set equal to the closest value inside the interval. It's important to note that the mathematical description should not be seen as exact relations between the nodes. It should rather be seen as a way of encoding the qualitative knowledge of the expert in a way that a computer can use.

The initial states of the different causal nodes are assumed to most likely be in the best possible state. The motor wear is assumed to have a very low chance of starting in a significantly worse state, while the other nodes have a significant chance of starting in a significantly worse state. This behavior is modeled with a doubly truncated normal distribution with zero mean where the variance defines how likely it is that the node is in a worse state. The resulting distributions are given in Table 5.5. Different distributions, such as the beta distribution, could have been used as well. The beta distribution has the advantage of naturally being within 0 and 1, thereby avoiding the truncation which is needed for the normal distribution. A drawback with the beta distribution is that its form is defined by shape parameters that the authors found less intuitive to work with than the variance of the normal distribution. This together with the fact that the truncated normal distribution has been previously used for modeling qualitative judgments [123] made the choice fall on the normal distribution.

All of the measurement nodes (Table 5.6), except those that are leaf nodes, are modeled as consisting of a normal distribution plus an exponential distribution. The normal distribution acts as a low-pass filter smoothing out noise. The exponential distribution biases the distribution towards measuring good values. This produces asymmetric behavior where the underlying state quickly changes when a poor state is measured, but changes slower when a good state is measured. This enables the system to react quickly to worsening conditions while still keeping conservative for a while when the situation seems to improve. Note that this produces a biased estimator as the mean measurement will be lower than the underlying causal factors. This bias is partly counteracted by adding its effect to the mean of the normal distribution and by considering the bias when evaluating the risk. This solution is not perfect, but as exact values are not needed to evaluate the risk it is deemed acceptable.

The *disturbance* and *environment unobservability* nodes (see Table 5.5) are modeled as slowly varying. The variation consists of a scaling factor that contains the state and added normally distributed white noise. The state of the *motor wear* can only get worse. A very small non-negative random trend is therefore added to the *motor wear* at each time step. The *varying motor effect* can suddenly jump up, this is modeled as a somewhat low probability of the state becoming any worse state with equal likelihood. The *varying motor effect* should also gradually decrease, this is modeled as a pure negative normally distributed white noise. This difference in the model for *motor wear* and *varying motor effect* ensures that short-lasting high motor loads are explained with the *varying motor effect*, while long-lasting effects are explained as *motor wear*.

Note that for nodes related to component health, such as *motor wear*, methods from reliability analysis [124] can be used to better model how the component will degrade over time. A very simple model is employed here to limit the scope of this article. Future work on including methods from reliability analysis in supervisory risk control would be of interest, perhaps in a case study with more focus on the health of different hardware components.

The continuous definitions given in this section are discretized using the tools available in GeNie [125]. GeNie can evaluate CPTs based on equations with specified discretization intervals. A tradeoff between computational burden and precision has to be made when choosing the number of discretization intervals. For this case study, 10 intervals were chosen. The measurement nodes and causal factors have uniformly distributed intervals. The scenario nodes have intervals that produce a

Table 5.3: The definition of scenario and loss nodes.

Name	State definition	Qualitative description	Function
L1) Damage to drone or facility	Value loss per time	The damage potential increases linearly with height, there is still a risk of damage at low heights. Full tether extension is 40m.	$0.8\,(0.1 + 0.3\,\mathit{Height\ over\ ground}/40)\,(S5 + S6 + S7)$
L2) Mission aborted	Value loss per time	The mission is always aborted if a loss of control occurs. The cost is substantially lower than the worst-case cost of damaging the drone.	$0.2\,(S5 + S6 + S7)$
S7) Excessive disturbance causes loss of controlled flight	Number of occurrences per time, [0-1]	Much more likely to occur with very high disturbance.	$(\mathit{Disturbance})^{30}$
S6) Motor failure causes loss of controlled flights	Number of occurrences per time, [0-1]	More likely to occur with high motor wear.	$(\mathit{Motor\ wear})^{3}$
S5) Physical contact with obstacle causes loss of control	Number of occurrences per time, [0-1]	Increasing the kinetic energy increases the chance of losing control. Control can still be lost at low speeds if the contact is uncontrolled. For scenario S2 a bit too small a safety distance will let the drone reduce the speed somewhat before impact.	$S4(0.3 + 0.1\,\mathit{Maximum\ speed}^2) + S3(0.2 + 0.1\,\mathit{Maximum\ speed}^2) + S2\,0.4\max(\mathit{Maximum\ speed} - \mathit{Effective\ safety\ distance/0.9},\,0)^2$
S4) Motor saturation causing deviation beyond safety distance	Number of occurrences per time, [0-1]	Much more likely to occur when total motor load is close to saturation. The frequency decreases with an increased safety distance. Deviation beyond safety distance can still occur, even though unlikely, at the largest safety distance.	$\max(1.1\,\mathit{total\ motor\ load} - 0.1 + \mathit{Maximum\ vertical\ acceleration}/10,\,0)^4\,(0.01 + 0.99\,(1.2 - \mathit{Effective\ safety\ distance})^2)$
S3) Disturbance causing deviation beyond safety distance	Number of occurrences per time, [0-1]	Frequency of occurrence increases quickly when the disturbance exceeds the safety distance. Worst-case disturbance requires a safety distance of around 0.7.	$(\mathit{Disturbance}^{0.5} - \mathit{Effective\ safety\ distance})^3$
S2) Safety distance smaller than braking distance	Number of occurrences per time, [0-1]	Through experimentation it was found that the braking distance was approximately linear with a coefficient of 0.9. The frequency should increase quickly when the safety distance is smaller than the braking distance.	$\max(0.9\,\mathit{maximum\ speed} - \mathit{Effective\ safety\ distance} + 0.6,\,0)^{10}$
S1) Presence of unobservable obstacles	Boolean, {0,1}	Frequency of occurrence increases when the environment unobservability increases.	$\mathcal{B}(\mathit{Environment\ unobservability}^{4.5})$

Table 5.4: The definition of intermediate nodes.

Name	State definition	Qualitative description	Function
Effective safety distance	Distance [0.1-1.2]	Equal 0 if there are unobservable obstacles present.	$S1$ *Safety distance*
Total motor load	Part of motor capacity, [0-1]	Motor load increases linearly with all factors.	*Motor wear* + *Varying motor effects* +*Motor load from turbulence* +*Motor load from tether*
Motor load from tether	Part of motor capacity, [0-1]	At the maximum height, the drone must lift the entire tether weight which uses 70% of the available motor capacity. When the drone is tilted the motor load of the drone can increase beyond what's expected from the height alone as the drone has to work against the tether tension. How much depends on a lot of unknown factors and is therefore uncertain.	$0.6\,$*Height over ground*$/40$ $+\max(\mathcal{N}(1,\ 0.3),\ 0)$ $($*Drone tilt*$\,-\,0.03)/0.1$
Motor load from tether	Part of motor capacity, [0-1]	Experiments show that the disturbance often causes a motor load with equal magnitude, but with a lot of variation.	$\max(\mathcal{N}(1, 0.3),\ 0)$ *Disturbance*

Table 5.5: Initial distribution and transition probabilities for causal nodes. $\bar{\mathcal{N}}(\mu, \sigma)$ represents the doubly truncated normal distribution where all values outside of [0,1] are omitted, and the resulting distribution is normalized. x_- represents the previous state of the same node.

Name	State definition	Initial distribution	Transition probability
Motor wear	Part of motor capacity, [0-1]	$\bar{\mathcal{N}}(0,\ 0.01)$	$x_- + \max(\mathcal{N}(0,\ 0.0001),\ 0)$
Varying motor effect	Part of motor capacity, [0-1]	$\bar{\mathcal{N}}(0,\ 0.6)$	$x_- - \max(\mathcal{N}(0,\ 0.1),\ 0)$ $+\mathcal{B}(0.05)\,\mathcal{U}(0,\ 1 - x_-)$
Disturbance	[0-1]	$\bar{\mathcal{N}}(0,\ 0.3)$	$\mathcal{N}(0.95\,x_-,\ 0.05)$
Environment unobservability	[0-1]	$\bar{\mathcal{N}}(0,\ 0.2)$	$\mathcal{N}(0.92\,x_-,\ 0.18)$

Table 5.6: Definition of measurement nodes. Note that the distribution of measurements nodes that also are leaf nodes have no effect and are therefore set to uniform distributions.

Name	State definition	Distribution
Height over ground	[0m - 40m]	$\mathcal{U}(0, 40)$
Drone tilt	[0.03rad - 0.1rad]	$\mathcal{U}(0.03, 0.1)$
Measured motor load	[1700rpm - 1949rpm]	$\big(\mathcal{N}(\textit{Total motor load} + 0.2, 0.1) - \mathcal{E}(1)\big)\ (1949 - 1700) + 1700$
Measured roll/pitch error	[0.002rad - 0.04rad]	$(\mathcal{N}(\textit{Disturbance} + 0.1, 0.1) - \mathcal{E}(1))\ (0.04 - 0.002) + 0.002$
Camera noise	{0,1}	$\mathcal{B}(0.5)$
Number of filtered away points	[0 points - 6 points]	$\begin{cases} \mathcal{U}(0,6) & \textit{Camera noise} = 1 \\ 6\ \big(\mathcal{N}(\textit{environment unobservability} + 0.1, 0.3) - \mathcal{E}(1)\big) & \textit{else} \end{cases}$

logarithmic scale. The upper bound of each interval for the scenario nodes is evaluated with the function $(100^{(i+1)/N} - 1)/99$ where $i \in \{0, ..., N-1\}$ is the index of the interval while N is the number of intervals. The first interval starts at 0 and the last one ends at 1. A problem when adding multiple discretized states, such as in the nodes *total motor load* and *motor load for tether*, is that the discretization uncertainty accumulates. To avoid this problem the states were rounded down to the lower end of the interval before adding. This ensured that the output of the sum was in its lowest possible state if all inputs were in their lowest possible state.

Control algorithm

Safety distance (D), *maximum speed* (S), and *maximum vertical acceleration* (A), were chosen as parameters the supervisory risk controller should control. These parameters were chosen as they were identified as greatly affecting the identified scenarios. The supervisory risk controller will only change the vertical component of the maximum acceleration as altering the horizontal component made the system less stable. The motor load needed to accelerate in the vertical direction is also much larger than in the horizontal direction, making the vertical direction more important. The following cost function was chosen:

$$\begin{aligned}\{D,S,A\} =& argmin\Big(a(D - D_{lb})/(D_{ub} - D_{lb}) \\ &+ b\big(1 - (S - S_{lb})/(S_{ub} - S_{lb})\big) \\ &+ 0.25\big(1 - (A - A_{lb})/(A_{ub} - A_{lb})\big)\Big) \\ s.t.\quad & E[L1(D,S,A)] + E[L2(D,S,A)] < 0.01\end{aligned} \tag{5.1}$$

The *safety distance* can take values between $D_{lb} = 0.1\,\text{m}$ and $D_{ub} = 1.2\,\text{m}$, *maximum speed* between $S_{lb} = 0.1\,\text{m/s}$ and $S_{ub} = 1.5\,\text{m/s}$, and *maximum vertical acceleration* between $A_{lb} = 0.5\,\text{m/s}^2$ and $A_{ub} = 2\,\text{m/s}^2$. The upper limit on *max-*

imum speed and *maximum vertical acceleration* were chosen based on what felt comfortable to fly with in the location where the experiments were done. The maximum *safety distance* was set to be sufficient for allowing the drone to brake in time when flying at 1.5 m/s. To ensure an acceptable risk level a constraint is introduced that requires that the risk of losses L1 and L2, which are evaluated with the DDN, must be under a maximum risk threshold. This threshold is tuned in to give good behavior.

In different parts of an operation, it can be preferable to either try to minimize the *safety distance* or maximize the maximum speed. The human operator can during flight set a target for *maximum speed* or *safety distance*. If a *maximum speed* target is set then the upper bound for *maximum speed* is set equal this target and the parameter a is set to 1 while b is set to 5 to prioritize a large *maximum speed*. If a *safety distance* target is set then the lower bound of the *safety distance* is set equal to this target and the parameter a is set to 5 while b is set to 1 to prioritize a small *safety distance*. Maximum acceleration is in all cases considered the least important.

The computational time needed to evaluate a DBN or a DDN increases with the number of time steps, as each new time step introduces a copy of each node. To get an acceptable computational time it is therefore essential to limit the number of time steps that need to be evaluated. To ensure that the network can be evaluated fast enough only 1.5-time steps are considered [126]. 1.5 time steps consist of, as shown in Figure 5.8, one full time step with all the nodes and a “half” time step with only the previous state of dynamic nodes. The general process of using 1.5 time steps is as follows:

- Evidence is inserted on time-step t
- The optimal parameter choice is evaluated at time-step t
- The updated (posterior) distribution of the dynamic nodes at time step t is used as the new prior distribution of the dynamic nodes at time-step $t-1$

The *safety distance* and *maximum speed* were discretized into 10 intervals, while the *maximum vertical acceleration* was only discretized into 5 intervals as there was no need for higher resolution in this state. Evaluating the 500 possible permutations with these discretization intervals took about 0.2 s with the 1.5-time-step approach by using the exact solver available in the SMILE engine[60]. It was therefore deemed feasible to perform an exhaustive search.

To prevent the states from switching too quickly a low pass filter was applied to the states when they switch from a more conservative to a less conservative value. This ensures that the states change quickly when needed to reduce the risk, but change slowly in the opposite case. The low-pass filter is then formulated as follows where $\bar{D}, \bar{S}, \bar{A}$ represent the filtered parameter choices.

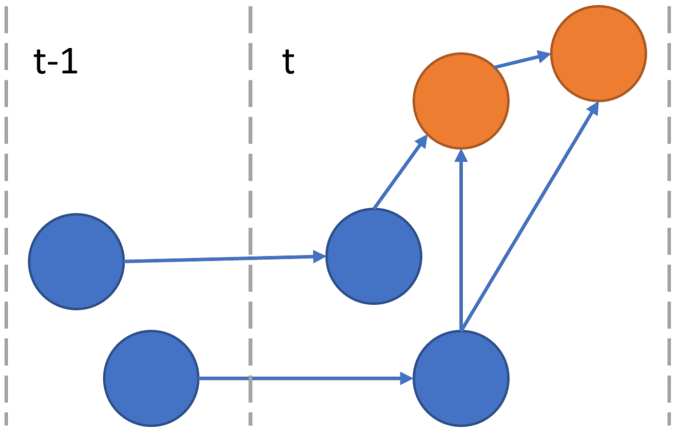


Figure 5.8: Example of considering 1.5 time steps in a DBN or DDN. Time step $t-1$ only contains the previous state of dynamic nodes.

$$\bar{D} = \begin{cases} D & D \geq \bar{D} \\ 0.15D + 0.75\bar{D} & D < \bar{D} \end{cases} \tag{5.2}$$

$$\bar{S} = \begin{cases} S & S \leq \bar{S} \\ 0.15S + 0.75\bar{S} & S > \bar{S} \end{cases} \tag{5.3}$$

$$\bar{A} = \begin{cases} A & A \leq \bar{A} \\ 0.15A + 0.75\bar{A} & A > \bar{A} \end{cases} \tag{5.4}$$

5.4 Results

This section presents the results of flying the drone with the supervisory risk controller active. The supervisory risk controller runs on a separate computer located on the ground, which communicates with the drone through the tether. The DDN is evaluated using the SMILE engine [60]. All tests were performed at the testing facility of ScoutDI shown in Figures 5.10, 5.14, 5.17 and 5.19.

Eight cases are considered that highlight the working of different nodes in the network. Case 1 and 2 show the effect of *disturbance*, Case 3 the *height over ground*, Case 4 *environment unobservability*, Case 5 *camera noise*, Case 6 *motor wear*, Case 7 *motor load from tether*, and Case 8 the combined effect of *environment unobservability* and *total motor load*.

Video of the experiments can be found at `https://youtu.be/RKhG9bguRJY` or by scanning 5.9.

Figure 5.9: Video of experiments. `https://youtu.be/RKhG9bguRJY`

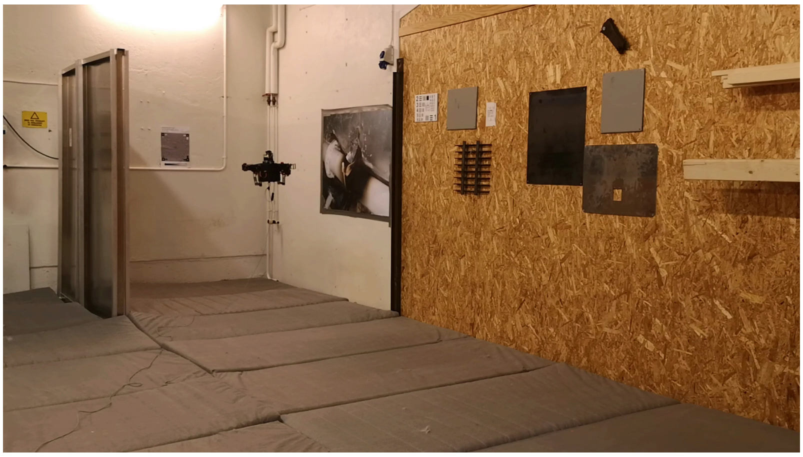


Figure 5.10: The ScoutDI drone flying in an enclosed area with more disturbance. Relevant for Case 1 and 3.

Case 1 - Entering an enclosed area

In this case, the drone enters an enclosed area, shown in Figure 5.10, where the walls interfere with the air streams produced by the drone causing significant disturbance. Figure 5.11 shows how the different states in the DDN develop over time. In the beginning the *safety distance* is set equal to its target value as there is no *disturbance*. The supervisory risk controller sets the *maximum speed* quite low as a higher value would cause the braking distance to be longer than the *safety distance*. As the drone enters the enclosed area around the 60 s mark, an increased *disturbance* is identified and the *safety distance* is increased to ensure that the disturbance will not cause the drone to collide. On the bottom plot of Figure 5.11 it is shown how the *total motor load* increases when the drone enters the enclosed area. Most of the increase is explained by the *motor load from disturbance*. *Max acceleration* is decreased to reduce the probability of motor saturation. Note that *environment unobservability* is non-zero even though there are almost no *filtered away points*. This is caused by the biased estimator that is used. The risk evaluation has taken this into consideration so that an *environment unobservability* of around 0.2 is considered a perfect condition.

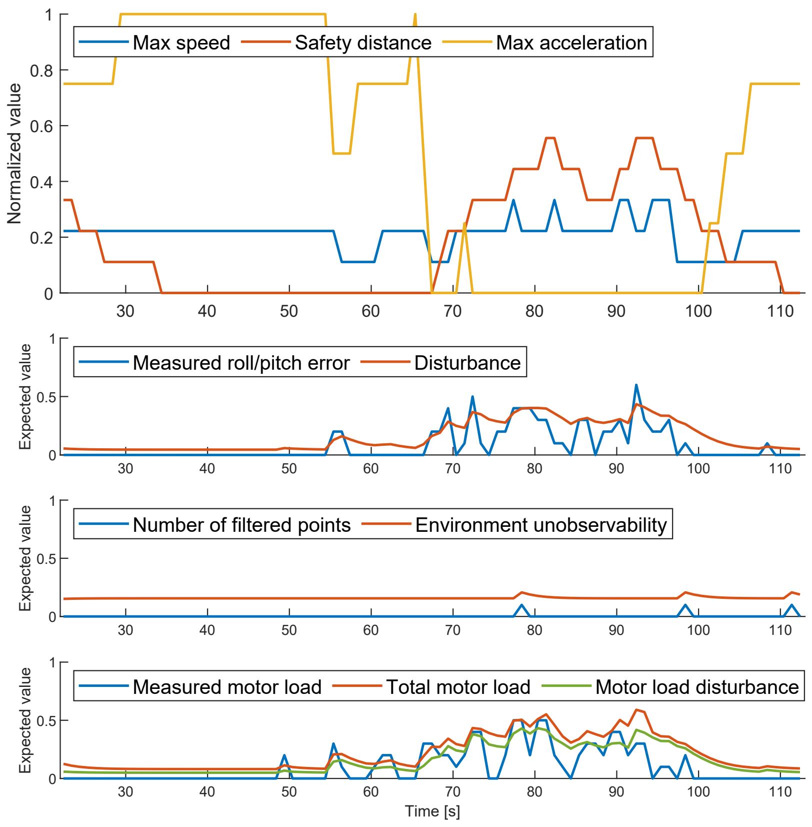


Figure 5.11: Case 1 - Entering an enclosed area with high disturbance. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *safety distance* target of 0.1 m (0 after normalization) is set.

Case 2 - Violently shaking the tether

Figure 5.12 shows a case where the tether is violently shaken, from a bit after the 206 s mark to a bit after the 214 s mark, thereby causing an excessive amount of *disturbance*. As there is a substantial chance of losing control due to this disturbance a mission abortion is recommended, which is marked with the red field.

Case 3 - High disturbance at different heights

Figure 5.13 shows a similar situation as in Case 1 but demonstrates the effect the height of the drone has on the choice of actions. At around the 130 s and the 150 s mark, the measured *height over ground* of the drone was artificially increased. Note that this artificial increase in height makes the *total motor load* increase far beyond the measured values, as a larger portion of the tether should have been suspended in the air thereby increasing the weight. This makes the supervisory risk controller act more conservative than needed. The *safety distance* can be seen to substantially increase when the *height over ground* increases, even though the

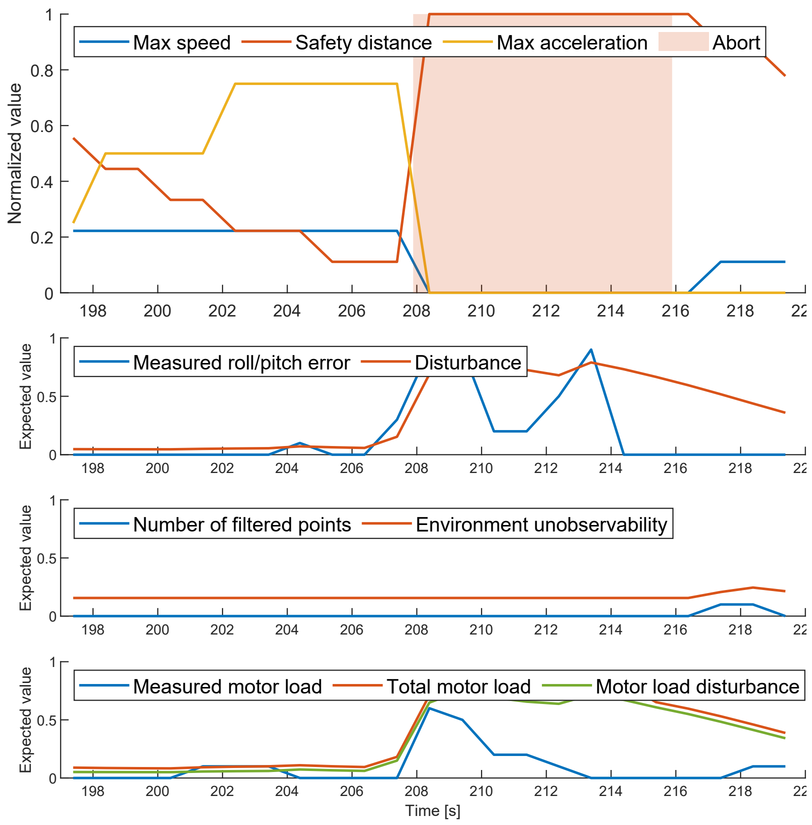


Figure 5.12: Case 2 - Shaking the tether causing excessive disturbance. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *safety distance* target of 0.1 m (0 after normalization) is set.

disturbance is similar or decreasing. This change is mainly caused by the increased consequences of flying at larger altitudes. Right after the 150 s mark mission abortion is recommended at a single time step. This is likely caused by a combination of the increased consequences caused by the increased height together with a slight increase in *environment unobservability* and *total motor load*.

Case 4 - Flying close to a wire fence

Figure 5.14 and Figure 5.15 show a case where the drone flies up to a wire fence that is hard to detect with the lidar. The sporadic reflections of the wire fence are filtered away by the dust filter. This makes the *environment unobservability* increase. As this renders the *safety distance* ineffective a speed reduction is instead performed so that the consequences of potential contact with an obstacle are reduced.

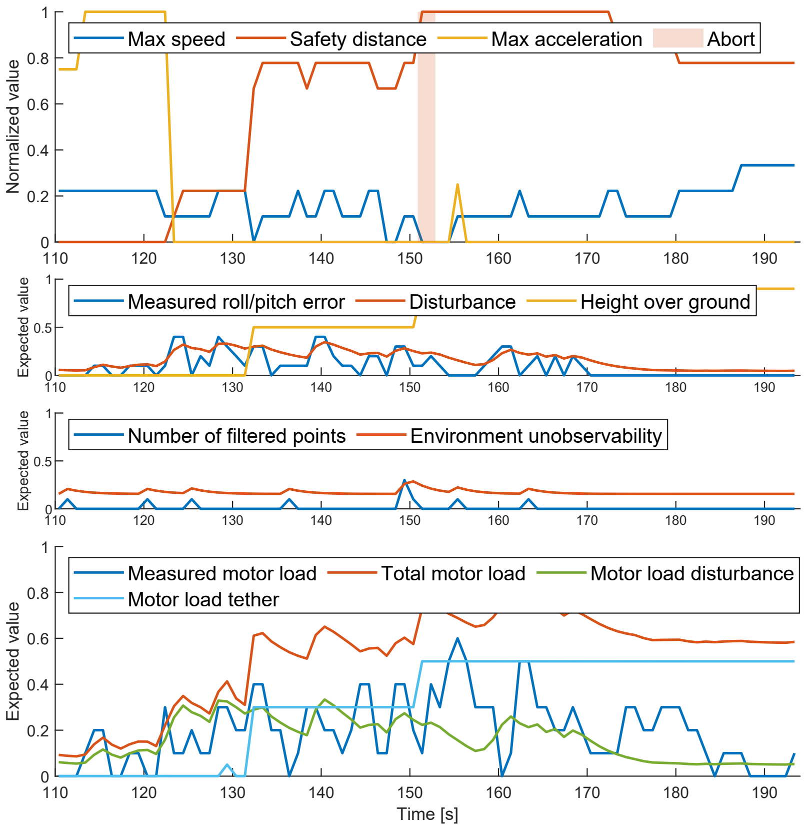


Figure 5.13: Case 3 - High disturbance at different heights. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *safety distance* target of 0.1 m (0 after normalization) is set.

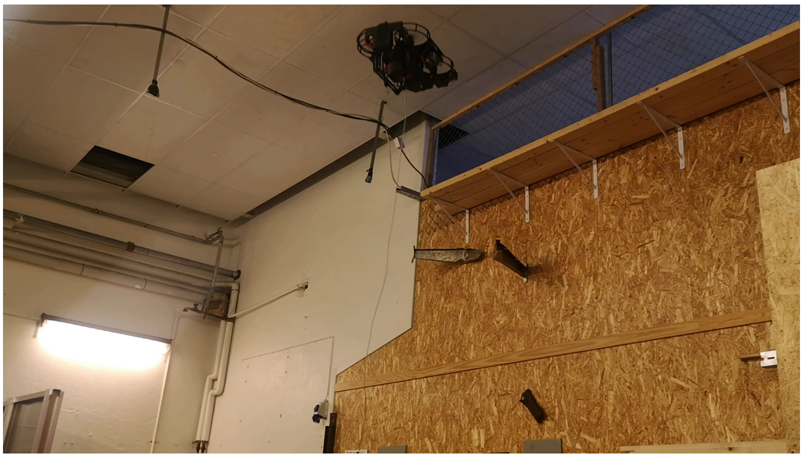


Figure 5.14: The ScoutDI drone flying close to a wire fence. Relevant for Case 4, 5, and 8.

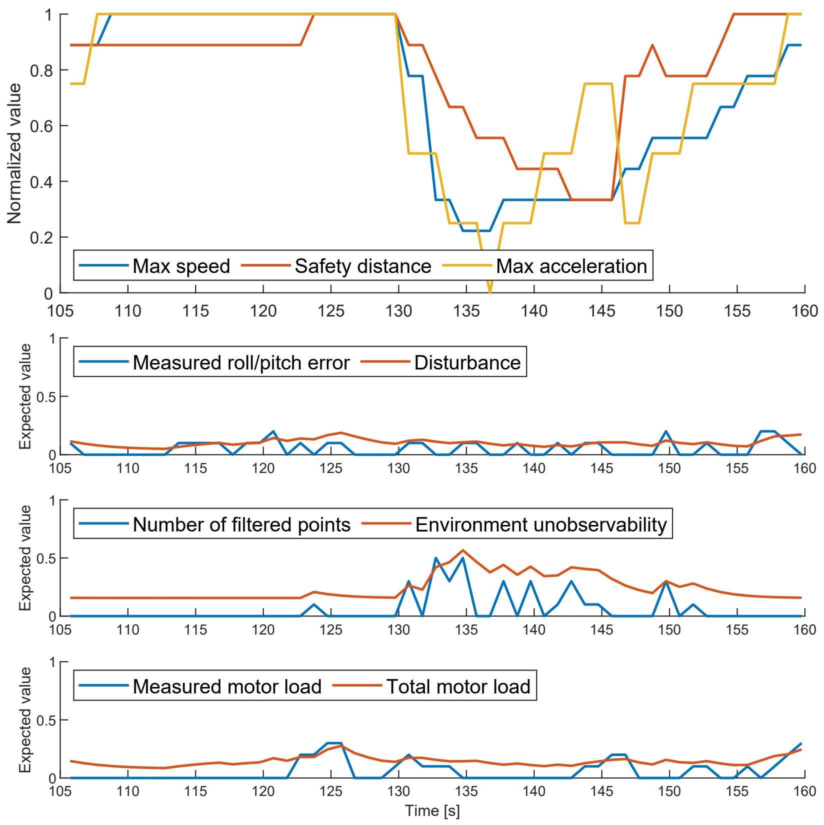


Figure 5.15: Case 4 - Flying close to a wire fence. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *maximum speed* target of 1.5 m/s (1 after normalization) is set.

Case 5 - Flying in dusty conditions

Figure 5.16 shows a case that mimics flying in dusty environments by flying the drone close to the same wire fence as in Figure 5.14 but the *camera noise* measurement is set to 1. This makes the *number of filtered away points* no longer give information on the *environment unobservability*, as it instead reports on actual dust in the environment. The *environment unobservability* then asymptotically increases towards its steady state value, unaffected by the *number of filtered away points*. This increase makes the supervisory risk controller reduce the *max speed*, but not as much as in Case 4.

Case 6 - Flying with extra weight

Figure 5.17 and Figure 5.18 show a case where a water bottle is fastened to the bottom of the drone which increases the *measured motor load*. A large *safety distance* is chosen to reduce the probability of hitting an obstacle if the motors saturate. Some of the motor load is explained by the increased disturbances caused by the

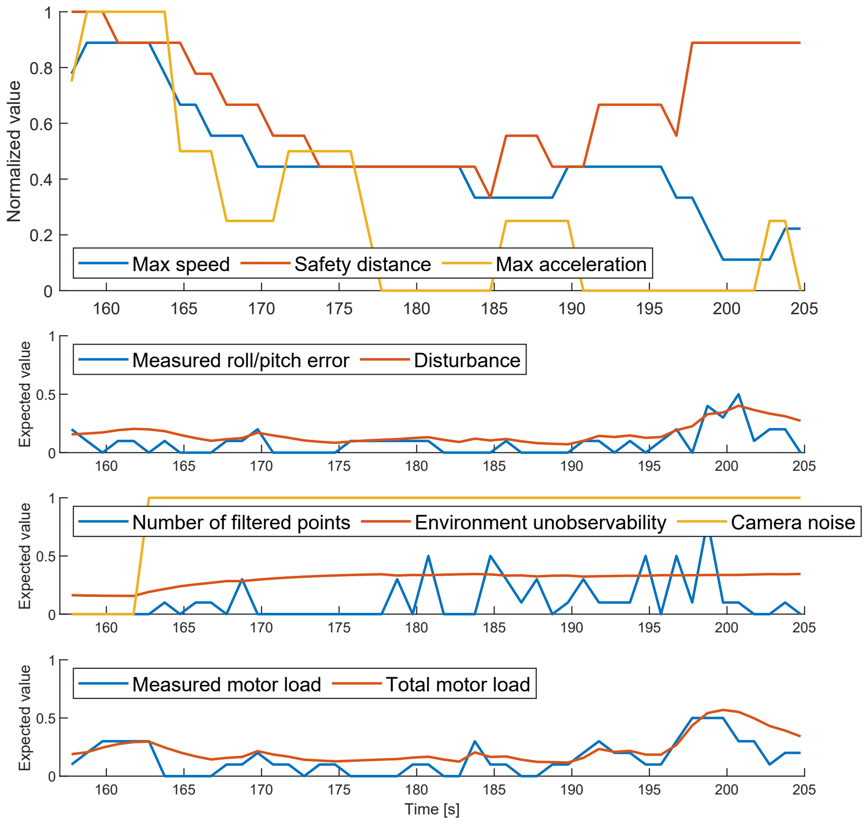


Figure 5.16: Case 5 - Flying in dusty conditions. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *maximum speed* target of 1.5 m/s (1 after normalization) is set.

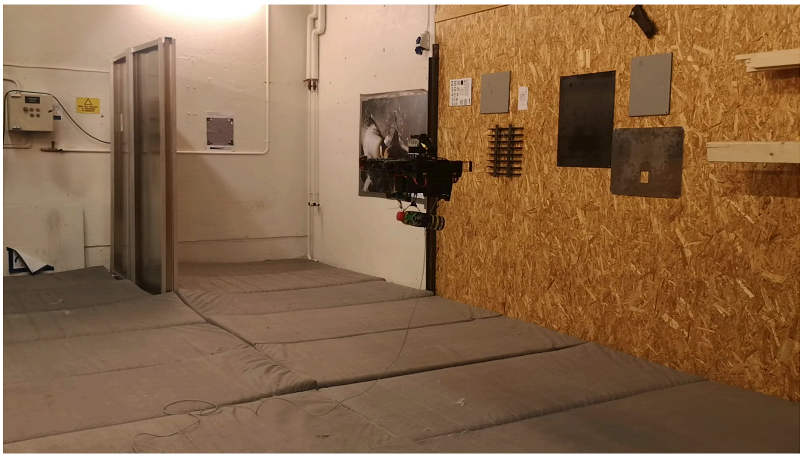


Figure 5.17: The ScoutDI drone flying with a water bottle fastened to it. Relevant for Case 6.

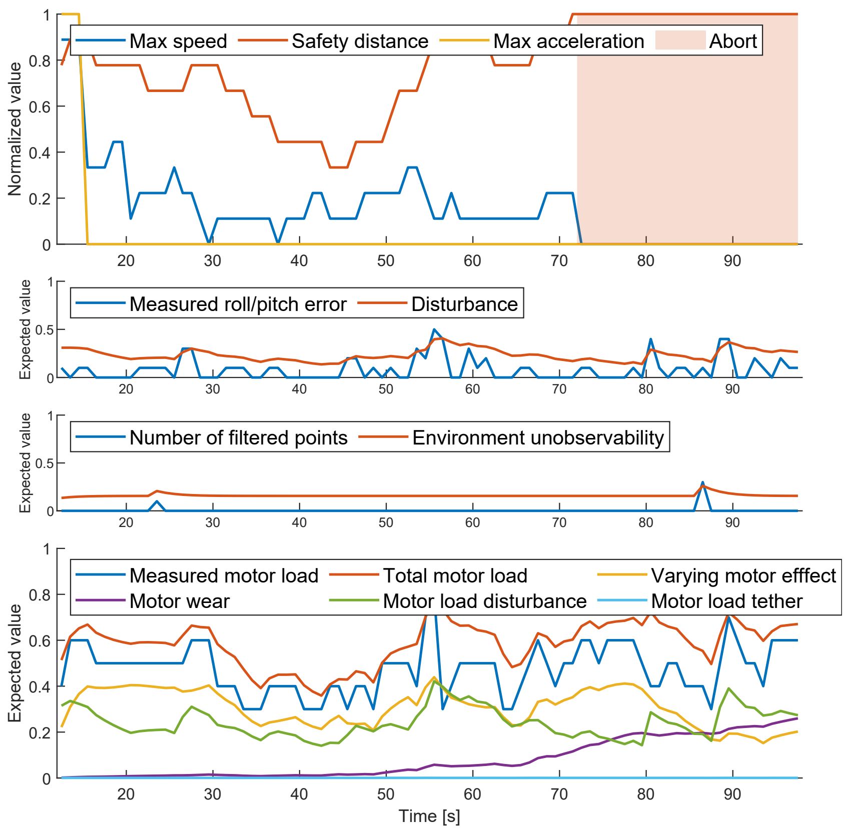


Figure 5.18: Case 6 - Flying with extra weight. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *safety distance* target of 0.1 m (0 after normalization) is set.

bottle swinging. As this is not sufficient to explain the entire effect the *varying motor effect* is also increased. As the *measured motor load* stays at a large value over time the probability that it is caused by *motor wear* is gradually increasing. Once it's sufficiently likely that *motor wear* is a problem then aborting the mission is recommended. As the drone is low over the ground and does not have a tilt the model evaluates that the extra motor load is not caused by the tether.

Case 7 - Flying with non-vertical tether load

Figure 5.19 and Figure 5.20 show a case where the water bottle is fastened to the tether which is lifted into the air. This produces a sideways force on the drone causing the drone to have a constant tilt. The measured *drone tilt* makes the *motor load from tether* increase significantly. The *motor wear* and *varying motor effect* do not change significantly when the *measured motor load* increases as the increase can be explained with the large *motor load from tether*. Note that a constant tilt of the drone has little to no effect on the *measured roll/pitch error*, as intended, as both the commanded and measured roll and pitch angles will quickly converge to the same offset. The constant tilt therefore only affects the *motor load from tether*.

Case 8 - Large motor load in front of the wire fence

Figure 5.21 shows a case where the drone flies in front of the wire fence shown in Figure 5.14 but where the *measured motor load* suddenly starts to increase. The increase is caused by the drone coming too close to the ceiling making it harder for the propellers to suck in air. In Case 6 the extra motor load was solved by increasing the *safety distance*. In this case, the large *environment unobservability* makes increasing the *safety distance* ineffective. In Case 4 the increased *environment unobservability* was solved by reducing the speed. This has little effect when hitting obstacles due to motor saturation. The supervisory risk controller recommends to abort the mission as there is no solution that can reduce the risk due to motor saturation when there is a large *environment unobservability*.

5.5 Discussion

Case 1, 4, and 6 demonstrate that the supervisory risk controller considers what the underlying causes are when finding a solution that ensures acceptable risk. 8 demonstrates that the supervisory risk controller has a holistic perspective when making decisions. Case 3 demonstrates that the supervisory risk controller not only considers the probability of occurrence but also the potential consequences when making a decision.

Case 5 and 7 demonstrate that the supervisory risk controller is able to combine information from multiple measurements to get a better understanding of the situ-

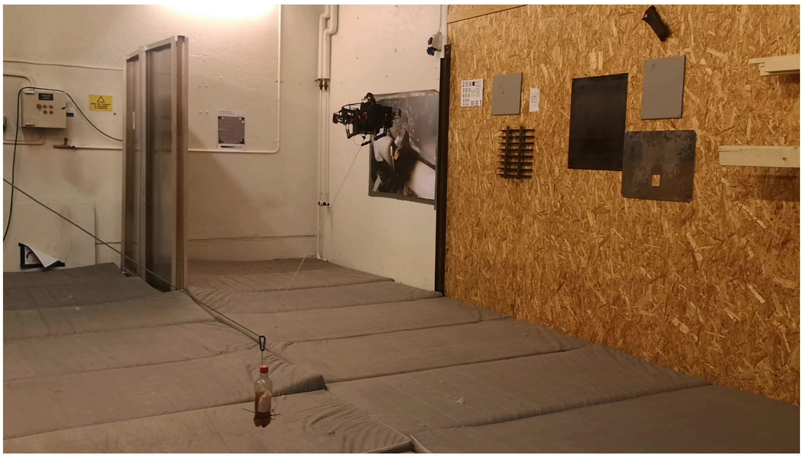


Figure 5.19: The ScoutDI drone flying with a water bottle fastened to an elevated tether. Relevant for Case 7.

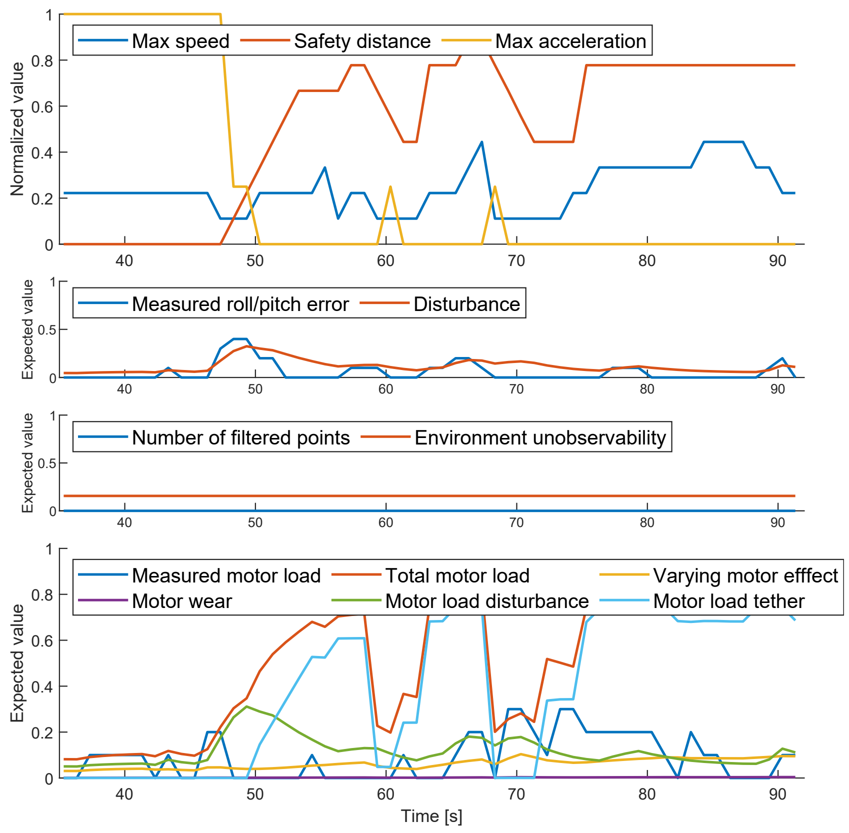


Figure 5.20: Case 7 - Flying with non-vertical tether load. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *safety distance* target of 0.1 m (0 after normalization) is set.

ation. 6 demonstrates that the supervisory risk controller can combine information over time to distinguish between different underlying causal factors.

Case 2, 6, and 8 demonstrate that interacting with the human operator, by requesting an abort, is crucial for ensuring a safe operation when there is nothing the supervisory risk controller can do to ensure an acceptable risk level.

Combined the different cases demonstrate that the system is able to understand the state of the underlying causes based on the available measurements and to make risk-based decisions based on this understanding.

Quantitatively evaluating the performance of high-level decision systems that act to minimize risk can be very difficult as there is no ground truth to compare with. As the purpose of the system is to support a human operator a true measure of quality would be to use the system for a while and see if it reduces the number of incidents or lets the human operator act less conservatively thereby increasing

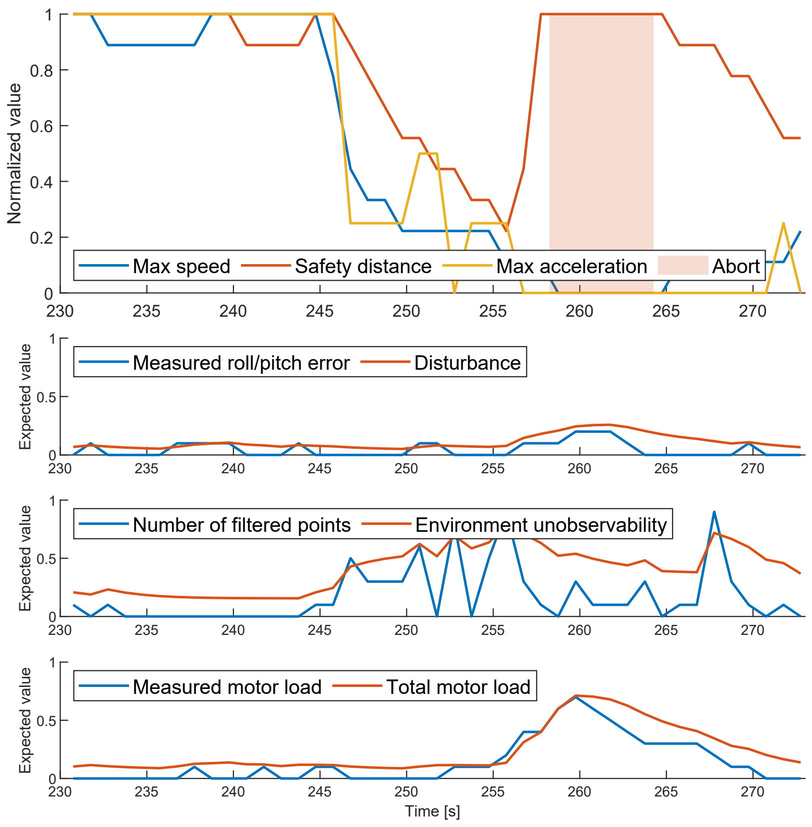


Figure 5.21: Case 8 - Large motor load in front of the wire fence. The expected value of relevant nodes normalized to be within 0 and 1 are shown. A *maximum speed* target of 1.5 m/s (1 after normalization) is set.

the performance of the system. Such an experiment is outside the scope of this chapter, and we are thus left with making an expert evaluation of whether actions made by the supervisory risk controller seem reasonable in the different cases. When demonstrating the system to employees at ScoutDI, a common comment was that beyond visual line of sight (such as in confined spaces) it can be hard to have a complete overview of the state of the drone, and in these cases having such a system would be very helpful. The supervisory risk controller can alert the human operator on what the underlying problems are thereby enabling the human operator to solve the problem. A common concern was that we had to ensure that the change in parameters would not prevent the human operator from getting out of dangerous situations, by for example reducing the *maximum vertical acceleration* when there is a strong updraft or error in the height estimator.

Performing a risk analysis definitely gave insight into which effects should be considered when building the risk model. One challenge with using the STPA is that it is not an iterative process thereby making the outcome sensitive to how the hazards are initially specified. Once loss scenarios are identified, there is no defined process for refining the scenarios to get more details. In the case study, starting with only the loss of controlled flight as a hazard, we got a scenario of contact with obstacles causing loss of controlled flight, with no obvious way of identifying the cause of the contact. This was handled by adding uncontrolled contact to the list of hazards, which had the uncommon effect of making a hazard the causal factor for a loss scenario. Different approaches for refining the scenarios identified with the STPA are presented in [127], [128].

The main challenge when developing the risk model is the quantification of the node states. There was too little historical data available to evaluate accurate probabilities and risk values. Instead, all the nodes were given unitless definitions, such as value loss per time, as seen in Table 5.3. This placed the focus on ensuring good ratios between the values of the different nodes.

Tuning in the right values to make the supervisory risk controller behave in a good manner was quite a large task. It took quite some testing to understand how different parameters affect the resulting behavior. Quantification of the DDN became simpler when using the proposed method of first giving a qualitative description and then encoding it as an equation. The fact that the nodes had meaningful states made the quantification of the DDN simpler as this gave some intuition on how the states should be compared to each other.

Bayesian networks have the advantage of supporting arbitrary probabilistic functions to model the relationships between nodes. This enabled this paper to use non-linear functions in the node definitions. A drawback is that the state spaces must be discretized, which can cause different types of discretization problems that have to be handled.

Modeling the distributions of the measurements as a combination of a normal distribution and an exponential distribution has the wanted effect of making the underlying state quickly react to worsening states while not reacting too fast to single good measurements. This is important for the case study as there are a lot of sporadic good measurements even in adverse situations. The normal distribution has the effect of smoothing out the noise. This is especially important in 1 as there are some sporadic non-zero measurements of *number of filtered away points* that would have caused the supervisory risk controller to be unnecessarily conservative if they were not filtered away. The bias introduced with this approach seems to be sufficiently counteracted as no effect was noticed on the resulting behavior.

An improvement that could be made to the system is to replace the conditional statement in the definition of *number of filtered away points* with a max statement. This would ensure that the *environment unobservability* does not increase even though the human operator has said that there is *camera noise* when there are no filtered away points. Another improvement would be to dynamically limit the upper bound of the *safety distance* such that it could never be larger than what fits in the drone's current environment.

The resulting behavior of the supervisory risk controller can, as with any behavior, instead be achieved by specifying an exhaustive list of how the drone should act for each possible combination of current and past measurements. Such solutions are practically impossible to implement for any but the simplest systems, as the list becomes inhibiting long. A more realistic alternative to the supervisory risk controller is therefore to implement simpler rules such that the safety margin should be a function of the disturbance and the max speed a function of the number of filtered away points. By instead using a probabilistic model as presented in this chapter it becomes possible to infer what underlying states are, thereby distinguishing between causal factors that cannot be directly observed, and to see different causal factors in light of each other thereby giving a holistic overview that is difficult to achieve with a short set of fixed rules. Additionally, by using a model of the system for decision-making, behavior can emerge that the developers of the system did not originally think of. This was the case for requesting aborting the mission when there were unobservable obstacles present at the same time as a large motor load. The authors had not considered this combination before it was accidentally observed during an experiment. This highlights another challenge with more direct rule-based solutions, as it is not trivial to ensure that the developers of such systems have thought about every possibility that must be considered.

5.6 Conclusion

This chapter develops and experimentally tests a supervisory risk controller for a tethered industrial inspection drone. An STPA is performed to identify relevant loss scenarios which are modeled with a DDN. The DDN is used to automatically set safety-critical parameters during operation to ensure safety.

From the experimental results, it is demonstrated how the DDN is able to identify the state of the environment and the drone itself by combining information from multiple measurements over time. Furthermore, it is shown how the DDN can be used to evaluate a set or parameters that ensures an acceptable risk level during operation and how it can be used to evaluate whether the mission should be aborted. Using a single DDN to model all causal factors and scenarios at once produces a holistic risk model. This enables the supervisory risk controller to consider multiple causal factors in light of each other when choosing an adequate set of parameters or deciding to recommend aborting the mission.

The resulting supervisory risk controller is used to assist unskilled operators during flight. However, the abilities demonstrated in this chapter are even more relevant for autonomous systems operating in unstructured environments without direct human monitoring or control. For such systems to safely operate it is essential to have a system, such as the supervisory risk controller, that monitors the operation and uses this information to modify how the system performs its tasks and to consider whether a task should be aborted.

Two main branches for further work are considered. Firstly, the proof of concept supervisory risk controller developed in this chapter could be extended to include more of the scenarios identified in the STPA. Especially scenarios relating to navigational state estimation and computational hardware should be explored further. Secondly, the general method could be applied to an autonomous operation, where, for example, the drone is tasked with surveying all surfaces in a confined space.

Acknowledgment

We would like to thank ScoutDI for their cooperation in this work. Specifically, we would like to thank Morten Fyhn Amundsen, Kristian Klausen, and Kristoffer Slåttsveen for their input to the risk analysis. Furthermore, we would like to thank the rest of ScoutDI for their help when preparing for the experiments and for lending us their equipment.

5.7 STPA scenarios

Table 5.7: Scenarios identified with the STPA. [U1] through [U4] refer to unsafe control actions introduced in 5.3.1 step A3

Scenario description	UCAs
Obstacles missing from the lidar scan preventing the collision avoidance algorithm from keeping the drone further away from the obstacle than the safety distance. Obstacles can miss from the lidar scan due to being thin enough making the laser beams miss or due to being transparent.	[U1] [U2]
A hard-to-detect obstacle shows sporadically up on the lidar scan but is filtered away by the dust filter (which aims at filtering away sporadic points caused by dust in the air), preventing the collision avoidance algorithm from avoiding the obstacle.	[U1] [U2]
Obstacle map containing two obstacles closer than the safety distance on opposite sides of the drone prevents the drone from moving away from either obstacle. This can occur either through the safety distance having increased, dynamic obstacles, or due to non-existing obstacles suddenly showing up on the obstacle map (due to noise, dust in the air, multi-path, or reflections from the drone itself that are not adequately filtered away).	[U2]
Obstacle map containing two obstacles closer than the safety distance on opposite sides of the drone can cause the drone to oscillate wildly between the two obstacles.	[U1]
Dynamic obstacles moving toward the drone faster than the drone's max-speed cause the drone to come closer to an obstacle than the safety distance.	[U2]
Joystick input commands the drone to move toward its blind zones. The drone cannot see obstacles below or above it due to the limited field of view of the lidar.	[U1]
Obstacle within the blind zone of the drone prevents the collision avoidance system from moving away from it.	[U2]
Lidar defect caused by overheating or other hardware failure causes no data to be gathered, thereby preventing the obstacle avoidance module from working.	[U1] [U2]
The obstacle avoidance module calculates a new input too slowly to fly the drone away from an obstacle in time.	[U1]
Lidar defect caused by overheating or other hardware failure prevents the drone from evaluating its speed and heading, causing the drone to not follow the velocity setpoint.	[U3]
Errors or displacement of IMU results in reduced quality of the navigation estimate, causing the drone to deviate beyond the safety distance or in the worst case lose control.	[U3], [U4]

An environment that lacks landmark features that can be tracked with a lidar prevents the navigation system from identifying the drone's speed and heading causing the drone to not follow the velocity setpoint.	[U3]
Too few points in the lidar scan prevent the navigation system from identifying the drone's speed and heading causing the drone to not follow the velocity setpoint.	[U3]
Motor saturation causes loss of a control degree of freedom, making the drone ignore velocity commands from the flight controller. Excessive motor saturation over a long time can in the worst case cause loss of controlled flight.	[U3], [U4]
Turbulence causes the drone to deviate from the velocity commands. Excessive turbulence can in the worst case cause loss of controlled flight.	[U3], [U4]
A long portion of the tether suspended in the air can start swinging from the winds produced by the propellers, causing a sideways thrust making the drone deviate from the velocity command. This can, in the worst case, cause a loss of controlled flight.	[U3], [U4]
Sudden large tether motion, such as it falling off an overhang, causes a sudden pull, making the drone deviate from the velocity command. This can, in the worst case, causes loss of controlled flight.	[U3], [U4]
Too low limits on the maximal linear acceleration cause the drone to deviate beyond the safety distance.	[U3]
Contact with obstacles at too high speed causes loss of controlled flight.	[U4]
Self-induced vibrations from motor or propeller damage cause loss of controlled flight.	[U4]
Motors or propellers are unable to produce adequate thrust to stabilize the drone due to wear or overheating.	[U4]
Power outage causing loss of controlled flight.	[U4]

Part II

Intention modelling for collision avoidance at sea

Chapter 6

Development of intention model

This chapter is based on the following publication

[51] **S. V. Rothmund**, T. Tengesdal, E. F. Brekke, and T. A. Johansen, "Intention modeling and inference for autonomous collision avoidance at sea," *Ocean Engineering*, vol. 266, p. 113 080, Dec. 2022. DOI: 10.1016/j.oceaneng.2022.113080

The method was developed by S. V. Rothmund with input from T. Tengesdal. Software development and simulations were done by S. V. Rothmund. Supervision was provided by E. F. Brekke and T. A. Johansen. The first draft was written by S. V. Rothmund and revised by T. Tengesdal, E. F. Brekke, and T. A. Johansen

6.1 Introduction

When navigating at sea, understanding the intentions of other ships can be crucial for avoiding accidents [129]. Blindly assuming that the other ship will follow the traffic rules put forth by COLREGs[13] is insufficient as shown in [130]. They demonstrated the existence of local unwritten rules and agreements between captains that went contrary to the rules specified by COLREGs. Furthermore, COLREGs is open to disagreements making it unsafe to act only based on your own interpretation of the situation [131], [132]. For an autonomous ship to safely operate in these conditions, it is essential that the ship can pick up on the intentions of other ships.

A large variety of ship collision avoidance algorithms exists in the literature [133], [134]. Most algorithms that consider COLREGs handle ships that do not fulfill the traffic rules by executing reactive evasive actions when the ships get close enough. In [135] this is handled by having a separate short-term controller, in addition to their COLREGs compliant controller, which disregards COLREGs when the ships are close enough. A different approach is taken in [14] where they have a separate collision risk and COLREGs compliance penalties. The collision risk penalty increases when the ships get closer, ensuring that an evasive action will be taken even if it conflicts with the main COLREGs rules.

A different approach is taken in [71] where they instead simulate multiple possible future trajectories the other ships can follow. The probabilities of the different trajectories are based upon the likelihood of the other ships having different intentions, such as being COLREGs compliant. This enables the collision avoidance algorithm to take early and substantial actions if the intentions are uncertain or if it becomes apparent that the other ship does not act according to the rules. However, [71] does not consider how these intentions can be identified.

Different methods exist for identifying the intentions of other ships [25], [132], [136]. [136] presents a method to identify whether the give-way ship is doing an evasive action or not. This enables the ship to comply with COLREGs rule 17, which states that stand-on ships should act if the give-way ship is not taking appropriate action. [25] presents a Bayesian model that evaluates the probability that the other ship follows its obligations as specified by COLREGs rules 14 to 17 based on its observed motion. [132] develop a scoring system to evaluate to what degree ships follow COLREGs rules 7, 8, and 13-17. This method is designed to evaluate different collision avoidance algorithms but can also be used online to evaluate how well other ships are acting in accordance with the rules.

These articles [25], [132], [136] evaluate whether the other ship is acting as expected based on the own-ships interpretation of the situation. They do not model the underlying causes making the ship not act as expected. These underlying causes could, for example, be a disagreement of the situation or one of the ships having priority over the other.

Works on intention modeling exist for air traffic [26], [137], road traffic [27], and for robot pedestrian interactions [28], [29]. These works show different ways of inferring the goal, behavior, or trajectories of the other agents in the encounter. Only [29] consider underlying causes that affect how an agent acts. They use information on whether a pedestrian is alone or in a group to affect the prior probability that it will hurry over at a flashing green light.

This chapter uses a DBN to model and infer the intentions of other ships in open waters. Different intention variables are defined based on the different ways ships can interpret and conflict with the behavioral rules specified by COLREGs. The DBN combines these intention variables with a model based on COLREGs Rule 7, 8, 11, and 13 to 18 to define the possible ways the ship can act. A ship's intentions are gradually inferred by ruling out all possible combinations of intention states that contradict the observed course and speed. This way of modeling ensures that the intention probabilities are independent of how often the model is updated with new observations.

The concept of intention inference can be compared with the concept of behavior consistency analysis used in fault diagnosis. Behavior consistency analysis concerns identifying faults in the system it runs on. This is done by observing input-output pairs and then identifying if they are best described with the nominal model of the system or with a model that describes the system with a particular fault [138]. Even though the intention model considered the inference of other agents, where only the output is observable, some similarities exist. Instead of inputs in the normal sense, we can consider the current situation (position, course, and speed of all vehicles) as the input. We then wish to evaluate whether a ship follows for example: the model of following COLREGS in the way we have interpreted it, the model of not complying with its give-way obligation and therefore keeping its course, or the model of disregarding the rules on how to give way. A substantial difference between behavior consistency analysis and the procedure presented in this chapter is that instead of explicitly making different models and then using hypothesis testing to identify which model that best explains our observation, we instead consider a single model that describes all the different possible behaviors. The inference capabilities supplied by hypothesis testing in fault diagnosis is here an integrated part of the model.

The contribution of this chapter and the novelty compared to earlier literature is a modeling framework that considers how underlying causes affect a ship's behavior and which can infer the state of multiple different intention variables based on measured properties. Modeling the underlying causes enables the model to identify situations that can cause misunderstandings, making it possible to take early actions to avoid a potentially dangerous situation. Furthermore, it enables the model to adapt to the current situation by letting additional information, such as relative ship size and location, affect the intentions. Being able to infer the state of multiple intention nodes enables the model to describe the future motion of other ships with higher fidelity than simply being COLREGs compliant or not. The resulting intention probabilities can be used for collision avoidance with algorithms that explicitly consider the intentions [71] or as decision criteria replacing the static distance used to decide when to always act to avoid collision [135].

The rest of the chapter is structured as follows. Section 6.2 presents the proposed DBN. Results of simulation trails are shown in Section 6.3. The results are discussed in Section 6.4 and a conclusion is given in Section 6.5.

6.2 Method

This section presents a DBN used to model and infer the intention of meeting ships. The term intentions will be used for a ship's internal states that we wish to infer such that we can understand how the ship will act. Examples of different intention variables are what the ship considers to be a safe distance, what priority it thinks it has relative to the other ships, and what it thinks that the COLREGs situation is.

The DBN model takes the perspective of a single ship, which will be called the reference ship, and models its relation to all other ships in the area. The index ρ

will be used to identify the other ships in the area. To model multiple ships, the model must be repeated for each ship. How to make inference using the model is described in Section 6.2.1.

Each of the intention variables are modeled as nodes in the DBN. These nodes are stochastic variables as the intention is unknown. The intention nodes are modeled as time-independent nodes as it is assumed that the intentions do not change within one encounter. The prior distribution of the intention nodes describes how often the different intentions are encountered in situations similar to this one. How these priors are designed is described in more detail in Section 6.2.4.

The intentions are updated based on different measured properties that can be evaluated based on the relative position between the ships, their course, and their speed. The different measured properties are given in Table 6.3. A tracking system is assumed to be used to evaluate the ships course, speed, and position. The tracking system is assumed to give high quality tracks, such that the intention module does not need to account for measurement uncertainty.

The DBN evaluates the probability that a particular combination of measurements and intention node states are compatible. Which combinations that are compatible are defined by COLREGs rules 8, and 13 to 18 and are described in Section 6.2.2 using logic statements. How these can be translated into CPTs is described in Section 6.2.3. The resulting DBN is shown in Figure 6.4.

When a new observation is made, the different measured properties are inserted as evidence on the measurement nodes in a new time-step of the DBN. These measurement nodes are time-dependent, thereby enabling the system to combine information over time. The network can be used to evaluate the probability that the observation is compatible with the prior distribution of the intention nodes. The distribution of the intention nodes can be updated by eliminating all combinations of intentions that contradict the observation. This is achieved by inserting evidence in the network stating that intentions and observed measurements are, in fact, compatible. The updated posterior distribution of the intention nodes can be used to give an updated prediction on how the reference ship will act. Two different ways of using the updated intention probability distributions for collision avoidance are outlined in Section 6.2.5.

Modeling whether a particular combination of observations and intention node states are compatible enables the system to gradually infer the reference ship's intentions without considering how often observations are given to the system. Giving the exact same observation multiple times to the system will not affect the probability distribution of the intention nodes, as the first observation has already eliminated all combinations of intentions that would be eliminated by the second observation.

A simplified example can illustrate this procedure. Figure 6.1 shows a simplified network that only considers when the ship will act. COLREGs rule 8(a) states

Table 6.1: Logical notation

Symbol	Meaning
$\wedge$	Logical and
$\vee$	Logical or
$\neg$	Logical not
$=$	Defines that the expressions on both sides are equal
$==$	Evaluates whether both sides are equal
$\neq$	Evaluates whether both sides are not equal
$\forall_{\rho \in P}$	Requires that an expression is true for all values of ρ in the set P
$\exists_{\rho \in P}$	Requires that there is at least one ρ in the the set P where the expression is true
$P \setminus \{\rho\}$	Considers the set P without the element ρ

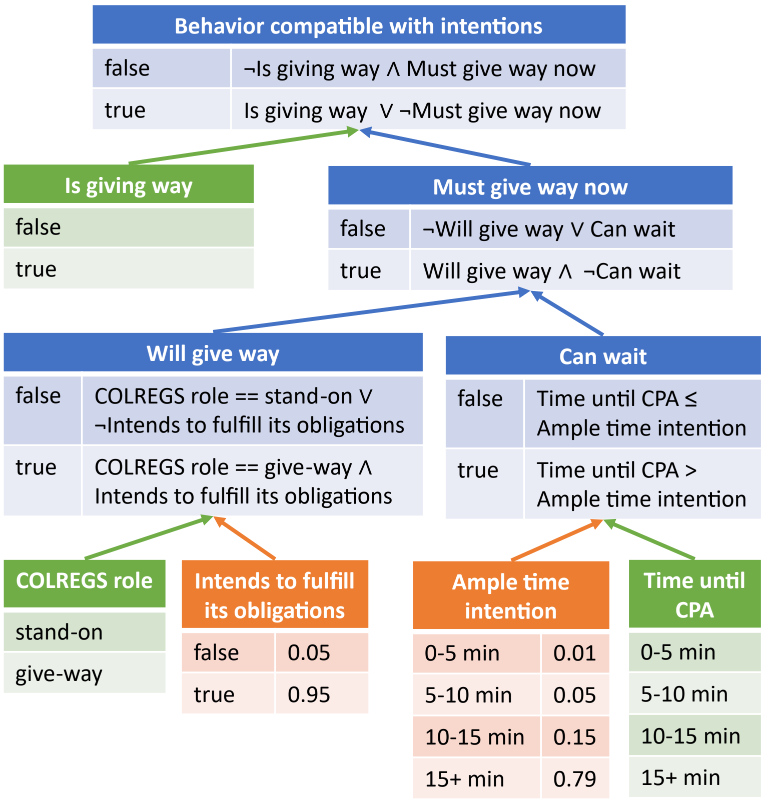

Behavior compatible with intentions
false
¬Is giving way ∧ Must give way now
true
Is giving way ∨ ¬Must give way now
Is giving way
false
true
Must give way now
false
¬Will give way ∨ Can wait
true
Will give way ∧ ¬Can wait
Will give way
false
COLREGS role == stand-on ∨ ¬Intends to fulfill its obligations
true
COLREGS role == give-way ∧ Intends to fulfill its obligations
Can wait
false
Time until CPA ≤ Ample time intention
true
Time until CPA > Ample time intention
COLREGS role
stand-on
give-way
Intends to fulfill its obligations
false 0.05
true 0.95
Ample time intention
0-5 min 0.01
5-10 min 0.05
10-15 min 0.15
15+ min 0.79
Time until CPA
0-5 min
5-10 min
10-15 min
15+ min

Figure 6.1: A simplified example network used to illustrate the proposed inference method. Measurement nodes are shown in green, intention nodes in orange, and modeling nodes in blue. The initial probability distribution is shown for the intention nodes.

that a ship should act in "ample time". Two intention nodes are then needed, one modeling the reference ship's definition of ample time and the other modeling whether the reference ship intends to follow this rule. When an observation is made, the following evidence is inserted: time until closest point of approach (CPA), which role the reference ship has according to COLREGs, and whether the reference ship is giving way. In this example, the observation is only compatible with the intention of the reference ship if either of the following is true: it is giving way, it has a stand-on (SO) role, if it does not intend to follow the rules, or if the time until CPA is longer than the reference ships definition of ample time.

The intention probabilities can be updated to reflect the observation by inserting evidence on the "Behavior compatible with intentions" node stating that it must be in the "true" state. If it, for example, is observed that the time until CPA is 10 minutes, the reference ship has a give-way (GW) role, and it is not giving way, then the model can exclude the possibility that the ship intends to follow its obligation to give way while at the same time considers ample time to be more than 10 minutes. It is left with the possibility that it will not follow its obligations at all or that it considers ample time to be shorter than 10 minutes. For this example, the updated probability that the reference ship does not intend to fulfill its obligations evaluates to 47%. This is due to the prior likelihood that the reference ship will give way at a short distance ($0.01 + 0.05 = 0.06$) is similar to the prior likelihood that it will not fulfill its obligations (0.05). This simplified example is unable to model the underlying causes that influence how a ship will act. The rest of this section handles this by considering many more of the COLREGs rules.

6.2.1 Basic procedure

For every new observation:

1. Insert information from observed position, course, and speed as evidence on the measurement nodes
2. Insert evidence stating that the compatible to all node (C) is in the state true.
3. Evaluate the updated probabilities for the different intention states
4. Expand the network with a new time-step

6.2.2 Intention model logic

This section presents a series of logic statements that define which combinations of intentions and observations that are compatible. These statements are based on the behavioral rules specified by COLREGs Rule 7, 8, 11, and 13 to 18. Rules regarding traffic separation schemes (Rule 10), narrow channels (Rule 9), and sailing vessels (Rule 12) are not considered in this chapter.

The section is structured following a top-down approach where the statement describing the most general model variable is presented first. Model variables that

Table 6.2: Intention variables

Symbol	Description	States
$\mathcal{I}_{AT}$	What time until CPA the reference ship considers ample time	real valued
$\mathcal{I}_{CC}$	Whether the reference ship intends to be COLREGs-compliant when performing evasive maneuvers	binary
$\mathcal{I}_{CS_\rho}$	What COLREGs situation the reference ship thinks it has towards ship ρ	$\{$*HO*, *OT-en* *OT-ing*, *CR-PS*, *CR-SS*$\}$
$\mathcal{I}_{GS}$	Whether the reference ship acts according to good seamanship	binary
$\mathcal{I}_{P_\rho}$	Whether the reference ship acts as if it has a lower or higher priority towards ship ρ	$\{higher, similar, lower\}$
$\mathcal{I}_{RC}$	What distance at CPA the reference ship considers a risk of collision	real valued
$\mathcal{I}_{RCF}$	How far in front of a ship the reference ship considers a crossing as risky	real valued
$\mathcal{I}_{SD}$	What the reference ship considers a safe distance at CPA	real valued
$\mathcal{I}_{SDF}$	How far in front of a ship the reference ship considers a crossing as safe	real valued
$\mathcal{I}_{SDM}$	What the reference ship considers a safe distance at CPA to the current midpoint (See Section 6.2.2).	real valued
$\mathcal{I}_{SS}$	At what distance the reference ship consider that the situation starts	real valued
$\mathcal{I}_{U}$	Whether the reference ship acts in an unmodeled way	binary

are used in more general statements are then gradually introduced. The different model variables are given in Table 6.4, intention variables in Table 6.2, measurement variables in Table 6.3, and parameters in Table 6.5.

$C[t]$ - Compatible to all

An observation is compatible with the intention states of the reference ship at time step t if it is compatible towards all ships in the area at that time step. The area considered must be large enough to encompass all ships that potentially affect how the reference ship acts. All observations are also considered compatible if the ship intends to act in an unmodeled manner ($\mathcal{I}_U$). This state works as a "dead hypothesis" that normally does not have any effect as discussed in [139], but springs into life under unexpected observations. These types of hypotheses enable a model to handle observations that do not fit with any of the nominal hypotheses, which in this case are behaviors that do not coincide with the behavioral rules outlined in this section. This state, therefore, works as a catch-all for behavior not

Table 6.3: Measurement variables. The values are evaluated based on the measured position, speed, and course of the different ships in an encounter. Measurements that cannot be directly evaluated based on the position, speed, or course are described when the measurement is first used in Section 6.2.2.

Symbol	Description	States
$\mathcal{M}_C[t]$	Current course of the reference ship	real valued
$\mathcal{M}_S[t]$	Current speed of the reference ship	real valued
$\mathcal{M}_{CS_\rho}[t]$	Current COLREGs situation reference ship has towards ship ρ (See Section 6.2.2)	$\{HO, OT\text{-}en\ OT\text{-}ing, CR\text{-}PS, CR\text{-}SS\}$
$\mathcal{M}_{D_\rho}[t]$	Current distance between the reference ship and ship ρ	real valued
$\mathcal{M}_{DCPA_\rho}[t]$	Distance between reference ship and ship ρ at CPA assuming both will keep their current course and speed	real valued
$\mathcal{M}_{DF_\rho}[t]$	How far the reference ship crosses in front on ship ρ assuming both keep their current course and speed. This value is set to ∞ if the ship does not cross in front of ship ρ	real valued
$\mathcal{M}_{DM_\rho}[t]$	Distance at CPA to the current midpoint between the reference ship and ship ρ, assuming constant course and speed for the reference ship. (See Section 6.2.2)	real valued
$\mathcal{M}_{MPS_\rho}[t]$	Whether the reference ship will pass the current midpoint between itself and ship ρ on its port or starboard side	$\{starboard, port\}$
$\mathcal{M}_{P_\rho}[t]$	Whether reference ship has passed ship ρ. (See Section 6.2.2)	binary
$\mathcal{M}_{TCPA_\rho}[t]$	Time until CPA between reference ships and ship ρ assuming both will keep their current course and speed	real valued
$\mathcal{M}_{AF_\rho}[t]$	Whether the reference ship will pass aft or in front of ship ρ assuming both keep their current course and speed. (See Section 6.2.2)	$\{Aft, Front\}$

Table 6.4: Model variables

Symbol	Description	States
$C[t]$	Observation compatible with the intentions of the reference ship	binary
$C_\rho[t]$	Observations and intentions compatible towards ship ρ	binary
$CEM_\rho[t]$	Correct evasive maneuver towards ship ρ	binary
$C\_CR\_SS_\rho[t]$	Correct crossing evasive maneuver with ship ρ on the starboard side	binary
$C\_CR\_PS_\rho[t]$	Correct crossing evasive maneuver with ship ρ on the port side	binary
$C\_HO_\rho[t]$	Correct head-on (HO) evasive maneuver towards ship ρ	binary
$C\_OT_\rho[t]$	Correct overtaking (OT-ing) evasive maneuver towards ship ρ	binary
$CIC_\rho[t]$	Change in course towards ship ρ	$\{starboard, straight\ port\}$
$CIS_\rho[t]$	Change in speed towards ship ρ	$\{higher, none, lower\}$
$GS_\rho[t]$	Good seamanship towards ship ρ	binary
$GWC_\rho[t]$	Gives way correctly towards ship ρ	binary
$IC_\rho[t]$	Initial course when the situation started towards ship ρ. Course is given in the NED frame.	real valued
$IS_\rho[t]$	Initial speed when the situation started towards ship ρ	real valued
$P_\rho[t]$	Has passed ship ρ safely	binary
$PA_\rho[t]$	There has been a port action towards ship ρ	binary
R_ρ	Role towards ship ρ	$\{GW, SO\}$
$RC_\rho[t]$	There is currently a risk of collision with ship ρ	binary
$RS_\rho[t]$	It is a risky situation towards ship ρ	binary
$SOC_\rho[t]$	Stands on correctly towards ship ρ	binary
$SD_\rho[t]$	The reference ship will cross at a safe distance towards ship ρ	binary
$SS_\rho[t]$	Situation has started towards ship ρ	binary
$SA_\rho[t]$	There has been a starboard action towards ship ρ	binary
$WGW_\rho[t]$	Will give way towards ship ρ	binary

Table 6.5: Example parameters chosen for demonstrative purposes. The parameters can be modified based on properties of the current situation, such as ship size, speed, and weather. The minimal acceptable definition of ample time (AT_{min}), safe distance at CPA (SD_{min}), safe distance front (SDF_{min}) and safe distance to the current midpoint (SDM_{min}), should be based on the maneuverability of the own-ship and how risk averse the operation should be.

Symbol	Description	Value
$\mathcal{P}_{CIC}$	Max change in course that is considered as keeping the course	10°
$\mathcal{P}_{CIS}$	Max change in speed that is considered as keeping the speed	2 m/s
AT_{min}	Ownships minimal accepted definition of ample time	60 s
SD_{min}	Ownships minimal accepted definition of safe distance at CPA	75 m
SDF_{min}	Ownships Minimal accepted definition of safe distance to cross in front	100 m
SDM_{min}	Ownships minimal accepted definition of safe distance to midpoint	75 m

considered by the DBN model.

Subscript ρ represents the ID of another ship in the encounter while P represents the set of all ships in the encounter other than the reference ship. Mathematically, this is expressed through the following logical clause:

$$C[t] = \left(\forall_{\rho \in P} C_\rho[t]\right) \lor \mathcal{I}_U \tag{6.1}$$

$C_\rho[t]$ - Compatible towards ship ρ

An observation is compatible with the intention states of the reference ship towards ship ρ if either of the following is true:

- The collision avoidance situation has not started yet (SS_ρ).
- The ships have passed each other safely (P_ρ)
- The ships will pass each other in such a manner that it is not a risky situation (RS_ρ).
- If the reference ship has a GW role (R_ρ) and gives way correctly towards ship ρ (GWC_ρ).
- If the reference ship has a SO role (R_ρ) and stands on correctly towards ship ρ (SOC_ρ).

$$\begin{aligned} C_\rho[t] =& \neg SS_\rho[t] \lor P_\rho[t] \lor \neg RS_\rho[t] \\ & \lor \Big((R_\rho == GW) \land GWC_\rho[t] \Big) \\ & \lor \Big((R_\rho == SO) \land SOC_\rho[t] \Big) \end{aligned} \tag{6.2}$$

$SS_\rho[t]$ - Situation started

According to COLREGs Rule 11, the behavioral rules only apply for ships in sight of each other. COLREGs Rule 3 specifies that a ship is in sight if it can be seen visually. At what distance the reference ship sees ship ρ is unknown and modeled with the intention variable $\mathcal{I}_{SS}$. The situation starts whenever the distance between the ships ($\mathcal{M}_{D_\rho}$) is shorter than the situation start intention. Map data can be used to evaluate at which distance the ships are likely to see each other.

$$SS_\rho[t] = SS_\rho[t-1] \vee \left(\mathcal{M}_{D_\rho}[t] < \mathcal{I}_{SS}\right) \tag{6.3}$$

$RS_\rho[t]$ - Risky situation

If there is a risk of collision (RC_ρ) at one point of time after the situation starts (SS_ρ), then the situation should be considered as risky.

$$RS_\rho[t] = \begin{cases} \mathit{false} & \mathit{if}\ \neg SS_\rho[t] \\ RC_\rho[t] \vee RS_\rho[t-1] & \mathit{otherwise} \end{cases} \tag{6.4}$$

$RC_\rho[t]$ - Risk of collision

Actions to avoid collision are only needed if the reference ship considers that there is a risk of collision (COLREGs Rule 7, 12, and 14). According to COLREGs Rule 7(i), a risk of collision exists if the compass bearing from the reference ship to ship ρ "does not appreciably change" [13]. How much change that is sufficient would depend on the distance between the ships, as one would experience a quicker bearing change once the ships get closer. To simplify this requirement, the expected crossing distance is used to evaluate whether there is a risk of collision. The acceptable distance when crossing in front can be larger than what is acceptable to the side of the ship. This is handled by defining two different intention variables, one specifying how far in front of a ship the reference ship considers it risky to cross ($\mathcal{I}_{RCF}$) and one specifying the distance at CPA that is considered risky ($\mathcal{I}_{RC}$). These are compared to the expected crossing distance in front ($\mathcal{M}_{DF_\rho}$) and at CPA ($\mathcal{M}_{DCPA_\rho}$) assuming both ships keep their current course and speed.

$$RC_\rho[t] = \left(\mathcal{M}_{DCPA_\rho}[t] < \mathcal{I}_{RC}\right) \vee \left(\mathcal{M}_{DF_\rho}[t] < \mathcal{I}_{RCF}\right) \tag{6.5}$$

$P_\rho[t]$ - Safely passed

If the reference ship has passed ship ρ ($\mathcal{M}_{P_\rho}$) and is at a safe distance (SD_ρ), then the reference ship does not need to consider the ship any longer. A ship is considered as passed if the time until closest point of approach, assuming constant course and speed for all ships, is negative.

$$P_\rho[t] = \mathcal{M}_{P_\rho}[t] \wedge SD_\rho[t] \tag{6.6}$$

$SOC_\rho[t]$ - Stands on correctly

The reference ship stands on correctly towards ship ρ if it does not change its course (CIC_ρ) or speed (CIS_ρ), or if it does a correct evasive maneuver (CEM_j) towards another ship (j) it has a GW role (R_j) for (Rule 17).

$$\begin{aligned} SOC_\rho[t] = &\Big(\big(CIC_\rho[t] == straight\big) \wedge \big(CIS_\rho[t] == none\big)\Big) \\ &\vee \Big(\exists_{\lambda \in P \setminus \{\rho\}} \big(R_\lambda == GW\big) \wedge CEM_\lambda[t]\Big) \end{aligned} \tag{6.7}$$

$GWC_\rho[t]$ - Gives way correctly

The reference ship gives way correctly towards ship ρ if it is executing a correct evasive maneuver CEM_ρ. According to COLREGs Rule 8, the ship must take evasive actions in what it considers "ample time" ($\mathcal{I}_{AT}$). The "time" in ample time is measured as the time until CPA assuming both ships keep their current course and speed ($\mathcal{M}_{TCPA_\rho}$). How long before CPA the reference ship consider as "ample time" is modeled with the intention variable $\mathcal{I}_{AT}$. The ship is allowed to stand on correct (SOC_ρ) before what it considers "ample time".

$$GWC_\rho[t] = CEM_\rho[t] \vee \Big(\big(\mathcal{M}_{TCPA_\rho}[t] > \mathcal{I}_{AT}\big) \wedge SOC_\rho[t]\Big) \tag{6.8}$$

$CEM_\rho[t]$ - Correct evasive maneuver

For an evasive maneuver to be correct, it must comply with "good seamanship" (GS_ρ) (COLREGs Rule 8) if the reference ship has an intention to act with "good seamanship" ($\mathcal{I}_{GS}$). Additionally, the maneuver must fulfill the requirements specified by COLREGs if the reference ship has an intention to be COLREGs-compliant when performing evasive maneuvers ($\mathcal{I}_{CC}$). COLREGs specify a set of situations and how to act in each scenario. These consist of overtaking (OT-ing) (Rule 13) another vessel, being overtaken (OT-en) (Rule 17), head-on (HO) (Rule 14), crossing with the other ship on the starboard side (CR-SS) (Rule 15), and crossing with the other ship on the port side (CR-PS) (Rule 17). What COLREGs situation the reference ship believes it has towards ship ρ is denoted as $\mathcal{I}_{CS_\rho}$.

$$\begin{aligned} CEM_\rho[t] = &\big(\neg\mathcal{I}_{GS} \vee GS_\rho[t]\big) \wedge \bigg(\neg\mathcal{I}_{CC} \vee \Big(\big((\mathcal{I}_{CS_\rho} == OT\text{-}en) \\ &\vee (\mathcal{I}_{CS_\rho} == OT\text{-}ing)\big) \wedge C\_OT_\rho[t]\Big) \\ &\vee \Big(\big(\mathcal{I}_{CS_\rho} == HO\big) \wedge C\_HO_\rho[t]\Big) \\ &\vee \Big(\big(\mathcal{I}_{CS_\rho} == CR\text{-}SS\big) \wedge C\_CR\_SS_\rho[t]\Big) \\ &\vee \Big(\big(\mathcal{I}_{CS_\rho} == CR\text{-}PS\big) \wedge C\_CR\_PS_\rho[t]\Big)\bigg) \end{aligned} \tag{6.9}$$

$SD_\rho[t]$ - Safe distance

According to COLREGs Rule 8, actions to avoid collision shall result in the ships passing at a safe distance. Whether the reference ship and ship ρ will pass at a safe distance is evaluated by assuming that both ships will keep their current course and speed. This assumption holds for ship ρ if it has a SO role, as SO ships are required to keep their course and speed (COLREGs Rule 17). If the reference ship has a GW role, then it is expected to mark its intent by substantially changing its course or speed (COLREGs Rule 8) before returning to the initial course. Assuming that it will keep its course and speed should result in passing at a safe distance if the ship has started to act to avoid a collision. As with risk of collision ($RC_\rho[t]$), different intention and measurement nodes are included for a safe crossing distance in front ($\mathcal{M}_{DF_\rho}$, $\mathcal{I}_{SDF}$) and at CPA ($\mathcal{M}_{DCPA_\rho}$, $\mathcal{I}_{SD}$).

$$SD_\rho[t] = \left(\mathcal{M}_{DCPA_\rho}[t] > \mathcal{I}_{SD}\right) \wedge \left(\mathcal{M}_{DF_\rho}[t] > \mathcal{I}_{SDF}\right) \tag{6.10}$$

$C\_OT_\rho[t]$ - Correct OT-*ing* evasive maneuver

COLREGs Rule 13 specifies that the OT-*ing* vessel shall keep out of the way of the vessel being "OT-*en*". Checking that the ships are crossing at a safe distance (SD_ρ) is therefore sufficient.

$$C\_OT_\rho[t] = SD_\rho[t] \tag{6.11}$$

$C\_HO_\rho[t]$ - Correct HO evasive maneuver

For HO situations, COLREGs Rule 14 specifies that the ships must make a starboard turn such that they pass each other port to port. As both ships have to give way in this situation, assuming that ship ρ will keep its current course is unrealistic. Instead, a new measurement is used that considers the distance at CPA to the current midpoint between the ships ($\mathcal{M}_{DM_\rho}$). As the current midpoint does not change when the ships' courses change, considering a safe distance to the current midpoint thereby requires that the reference ship has to do an evasive maneuver even though ship ρ has already changed its course. The distance at CPA to the current midpoint is evaluated assuming the reference ship will keep its current course and speed. Which distance to the midpoint the reference ship considers as safe is denoted as $\mathcal{I}_{SDM}$. Which side the reference ship passes with the midpoint on is denoted as $\mathcal{M}_{MPS_\rho}$.

$$C\_HO_\rho[t] = \left(\mathcal{M}_{MPS_\rho}[t] == port\right) \wedge (\mathcal{M}_{DM_\rho}[t] > \mathcal{I}_{SDM}) \tag{6.12}$$

$C\_CR\_SS_\rho[t]$ - Correct crossing starboard-side evasive maneuver

In a crossing situation, Rule 15 of COLREGs specifies that a ship should, in addition to cross at a safe distance (SD_ρ), avoid crossing in front of another ship it has on its starboard side. Whether the reference ship crosses aft or front of ship ρ

($\mathcal{M}_{AF_\rho}$) is evaluated by first finding the intersection point of the paths followed by the ships assuming that they keep their current course. Which ship that first arrives at this point crosses in front of the other.

$$C\_CR\_SS_\rho[t] = \big(\mathcal{M}_{AF_\rho}[t] == \mathit{aft}\big) \wedge SD_\rho[t] \tag{6.13}$$

$C\_CR\_PS_\rho[t]$ - Correct crossing port-side evasive maneuver

If a ship with the other on its port side is forced to take action, then COLREGs Rule 17(c) specifies that it, in addition to cross at a safe distance (SD_ρ), should avoid changing its course (CIC_ρ) towards port.

$$C\_CR\_PS_\rho[t] = \big(CIC_\rho[t] \neq \mathit{port}\big) \wedge SD_\rho[t] \tag{6.14}$$

$\mathcal{M}_{CS_\rho}[t]$ - COLREGs situation

According to COLREGs Rule 13(b), a ship is *OT-ing* another when it is coming up on the ship "from a direction more than 22.5 degrees abaft her beam" [13]. Uncertainty in the heading of the other ship can lead to different interpretations of the situation. Uncertainty in whether it is an *OT-ing* situation is modeled by using the classifier as shown in Figure 6.2. The size of the uncertainty region can be based on a combination of historical data and expert opinion. Different situations could be presented to different experienced captains where they could express their trust that other ships would identify this situation correctly. The values used in this chapter are chosen for demonstrative purposes.

A *HO* situation is defined by COLREGs Rule 14(a) to be when two vessels are meeting on "nearly reciprocal courses", while Rule 14 (b) specifies when a *HO* situation exists based on the visibility of different lights of the other ship. This opens up for disagreements from different definitions of "nearly reciprocal" and how the ships observe each other. With the presence of current and winds, a ship observing the course of the other by radar or AIS might come to a different conclusion than one observing the heading of the other ship based on the visibility of lights [132]. Furthermore, measurement uncertainties in the course of the other ship can lead to misunderstandings. The classifier shown in Figure 6.3 is used to accommodate this uncertainty. Identifying the uncertainty and mean of which angle a *HO* situation starts can be evaluated similarly to the *OT-ing* case. In addition, the mean can be chosen based on case law and certifying agency requirements as proposed in [132].

The probability that the reference ship evaluates the current situation as an *OT-ing* or *HO* situation is based on the two classifiers given in Figure 6.2 and Figure 6.3. The remaining probability gives the probability that the reference ship evaluates the situation to be a crossing situation. Whether the reference ship is in front or aft of the other ship when the situation starts defines whether it is *OT-ing* or being *OT-en*. Whether the other ship is on the port or starboard side defines whether

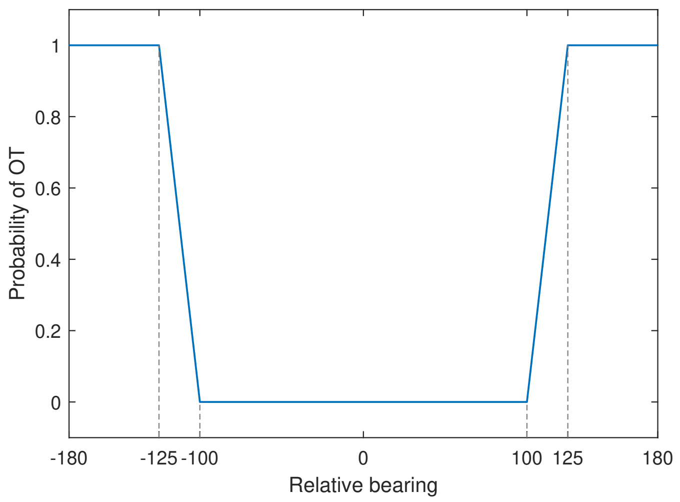


Figure 6.2: Classifier giving the probability that it is an *OT-ing* situation. Relative bearing is defined as the bearing from the ship being *OT-en* to the *OT-ing* ship relative to the heading of the ship being *OT-en*. 22.5° abaft the beam as specified in COLREGs Rule 13 is the same as ±112.5° relative to the heading. This classifier considers a 15° uncertainty in the situation.

it is a *CR-PS* or *CR-SS* situation. This information is inserted as virtual evidence on the measured COLREGs situation node, $\mathcal{M}_{CS_\rho}$.

According to COLREGs Rule 13(d), subsequent alterations in bearing do not change the situation. The situation is therefore defined when the situation starts, which can lead to misunderstandings as the different ships may define that the situation starts at different time points [131]. To model the uncertainty caused by when the reference ship thinks that the situation starts, a situation measurement node ($\mathcal{M}_{CS_\rho}$) is introduced. The state of this node is equal to the state of the situation intention node ($\mathcal{I}_{CS_\rho}$) only at the time-step where the reference ship thinks that the situation starts. At all other time-steps, the probability of measuring the different states of the measurement node is unaffected by the state of the intention node. Which time-step the reference ship thinks that the situation starts is uncertain, making it uncertain which measurement that defines the intention state. There should be an equal probability of measuring all states when the measurement node is independent of the intention node.

$$\mathcal{M}_{CS_\rho}[t] = \begin{cases} \mathcal{I}_{CS_\rho} & \textit{if } SS_\rho[t] \land \neg SS_\rho[t-1] \\ [0.2, 0.2, 0.2, 0.2, 0.2] & \textit{otherwise} \end{cases} \tag{6.15}$$

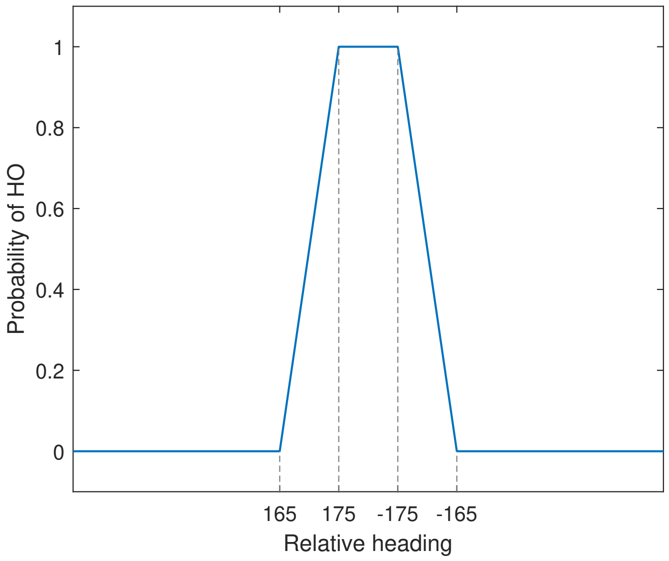


Figure 6.3: Classifier giving the probability that it is a *HO* situation. The relative heading between the two ships defined the probability. This classifier considers a 10° uncertainty in the situation.

R_ρ - Role

A ship must give way if it has lower priority ($\mathcal{I}_{P_\rho}$), either as specified in COLREGs Rule 18 or due to unwritten rules [130]. If the ship has higher priority, it must stand on. If the priority is similar, then the role is given by Rule 13 to 15. In a *HO* situation, both ships must give way (Rule 14). In an *OT-ing* situation, the *OT-ing* vessel must give way (Rule 13). In a crossing situation, the one with the other ship on its starboard side must give way (Rule 15).

$$R_\rho = \begin{cases} GW & \begin{array}{l} \textit{if } (\mathcal{I}_{P_\rho} == \textit{lower}) \vee \Big((\mathcal{I}_{P_\rho} == \textit{similar}) \\ \quad \wedge \big((\mathcal{I}_{CS_\rho} == HO) \vee (\mathcal{I}_{CS_\rho} == CR\text{-}PS) \\ \quad \vee (\mathcal{I}_{CS_\rho} == OT\text{-}ing) \big) \Big) \end{array} \\ SO & \textit{otherwise} \end{cases} \tag{6.16}$$

$GS_\rho[t]$ - Good seamanship

Good seamanship is difficult to define and can contain many different behaviors. In this work, good seamanship restricts the ship from changing which side it turns towards to avoid collision. The ship is not allowed to have made both a starboard action (SA) and a port action (PA) during a collision encounter.

$$GS_\rho[t] = \neg\big(SA_\rho[t] \wedge PA_\rho[t]\big) \tag{6.17}$$

$$SA_\rho[t] = \begin{cases} \mathit{false} & \mathit{if}\ \neg SS_\rho[t] \\ \big(CIC_\rho[t] == \mathit{starboard}\big) \vee SA_\rho[t-1] & \mathit{otherwise} \end{cases} \tag{6.18}$$

$$PA_\rho[t] = \begin{cases} \mathit{false} & \mathit{if}\ \neg SS_\rho[t] \\ \big(CIC_\rho[t] == \mathit{port}\big) \vee PA_\rho[t-1] & \mathit{otherwise} \end{cases} \tag{6.19}$$

$CIC_\rho[t]$ - Change in course

A change in course is evaluated by comparing the initial course (IC_ρ) with the measured course ($\mathcal{M}_C$). The initial course is saved when the situation starts (SS_ρ). If the change in course is less than $\mathcal{P}_{CIC}$ then it is considered as keeping the course. $\mathcal{P}_{CIC}$ should be chosen small enough to ensure that all intended course changes are marked as such, while being large enough to ensure that measurement uncertainty and small oscillations due to waves are not marked as a course change.

$$IC_\rho[t] = \begin{cases} \mathcal{M}_C[t] & \mathit{if}\ \neg SS_\rho[t] \\ IC_\rho[t-1] & \mathit{otherwise} \end{cases} \tag{6.20}$$

$$CIC_\rho[t] = \begin{cases} \mathit{starboard} & \mathit{if}\ \mathcal{M}_C[t] > \big(IC_\rho[t] + \mathcal{P}_{CIC}\big) \\ \mathit{port} & \mathit{if}\ \mathcal{M}_C[t] < \big(IC_\rho[t] - \mathcal{P}_{CIC}\big) \\ \mathit{straight} & \mathit{otherwise} \end{cases} \tag{6.21}$$

$CIS_\rho[t]$ - change in speed

The initial speed (IS_ρ) and change in speed are evaluated in the same manner as for the course. The same considerations should be made when choosing $\mathcal{P}_{CIS}$.

$$IS_\rho[t] = \begin{cases} \mathcal{M}_S[t] & \mathit{if}\ \neg SS_\rho[t] \\ IS_\rho[t-1] & \mathit{otherwise} \end{cases} \tag{6.22}$$

$$CIS_\rho[t] = \begin{cases} \mathit{higher} & \mathit{if}\ \mathcal{M}_S[t] > \big(IS_\rho[t] + \mathcal{P}_{CIS}\big) \\ \mathit{lower} & \mathit{if}\ \mathcal{M}_S[t] < \big(IS_\rho[t] - \mathcal{P}_{CIS}\big) \\ \mathit{none} & \mathit{otherwise} \end{cases} \tag{6.23}$$

6.2.3 Translation into DBN

A DBN is made from the logic statements given in Section 6.2.2. A node is introduced for each intention variable, measurement variable, and model variable. Arcs are introduced based on the dependencies given by the equations in Section 6.2.2. The resulting topology can be seen in Figure 6.4.

The logical statements given in Section 6.2.2 need to be translated into CPTs to be used by the DBN. This can be done by evaluating whether the output is “true”

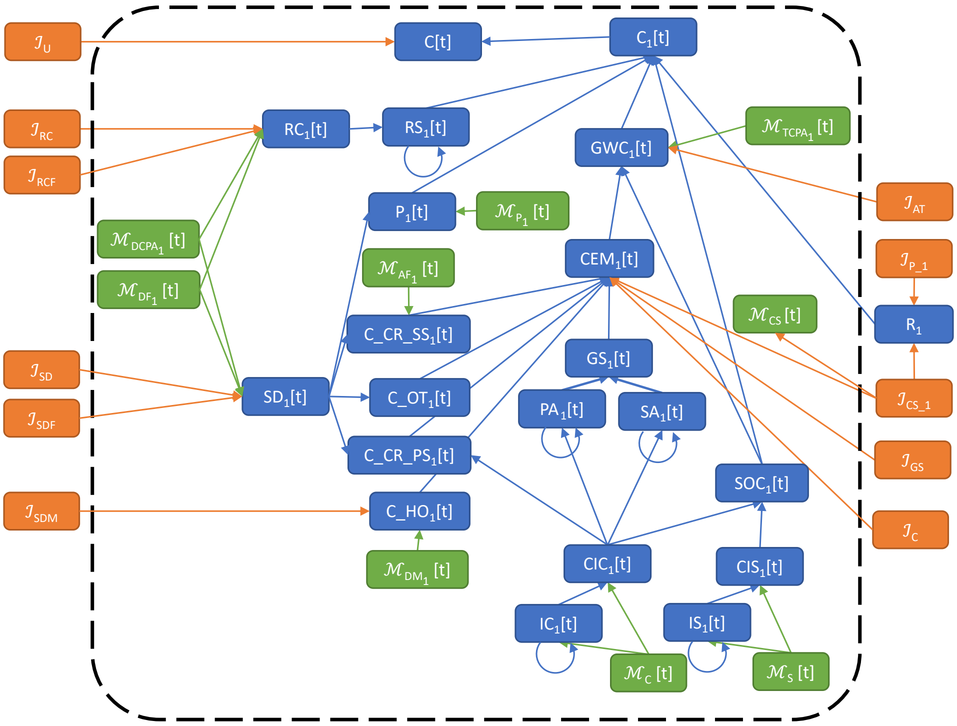


Figure 6.4: Figure showing the topology of the resulting DBN for a single ship encounter. Nodes related to situation start ($\mathcal{I}_{SS}$, $SS_\rho[t]$) are omitted to reduce complexity. See Tables 6.2 to 6.4 for abbreviations. Subscript 1 indicates that this model considers the relation between the reference ship and ship with index 1. In a multi-ship encounter, all nodes with index subscript would be repeated for any additional ship in the encounter. Green nodes represent measurements, orange node intentions, and blue nodes model variables. All nodes inside the dashed box are time-dependent and are repeated for each time step. Circular arrows indicate connections between subsequent time steps.

or "false" for all combinations of inputs. This results in CPTs consisting of 0/1 probabilities. Nodes that according to Tables 6.2 to 6.4 are real-valued must be discretized. A suitable range and discretization step must be defined. The software GeNIe [125] allows the user to specify equations and to use real-valued nodes. It can then automatically discretize and translate these equations into CPTs.

6.2.4 Priors

Information from the current situation, such as ship types and the type of environment, can improve the prior distributions of the intention nodes. Examples of different factors that could be considered are shown in Table 6.6. These influencing factors can be included as time-independent nodes that affect the intention states.

Different approaches can be followed to identify factors that affect the intentions. One way is to have a workshop with experts in the field, such as experienced

Table 6.6: Factors that can influence the intentions of the reference ship. Table 6.7 specifies and quantifies the dependencies.

Factor	Reason	States
Maneuverability	A poor maneuverability requires earlier actions and larger margins	low/medium/high
Location	Ships tend to act earlier and have larger margin in open seas than inland waterways	open sea/innland
Ship type	A leisure craft is less likely to know and follow rules and best practice	commercial/ leisure
Relative ship size	Larger ships tends to have priority over smaller ships [130]	smaller/similar/ larger
Speed	Ships require larger safety margins when going at a fast speed	slow/fast

captains. This workshop can be similar to risk analysis workshops such as in [140] and [119]. Another option is to study captains during operation as done in [130]. This has the advantage of being more correct than a workshop, but some factors might not show up during the study. A last option is to analyze historical data logged with the AIS that larger vessels are required to be equipped with [141]. This method would be more general as much more data from different ships and situations could be analyzed. It will, however, be limited to the information that is logged with the AIS, which does not necessarily include all factors that could be of interest. A combination of the three approaches is preferable to maximize correctness and completeness.

The same methods can be used for quantifying how the intention nodes are affected by the identified factors. AIS data could be used to build prior distributions on, among others, how far before CPA different types of ships tend to give way and how close they tend to be at CPA. This information could be supplemented with data from operation studies and expert judgment to model how factors not included in the AIS affect the distribution. Different methods for building CPTs based on expert information are analyzed in [100].

Performing a thorough identification and quantification is outside the scope of this chapter. Table 6.7 shows the quantification used to produce the results presented in Section 6.3.

6.2.5 Using the intentions

This section presents two different ways of using the evaluated intention probabilities for collision avoidance.

Table 6.7: Prior probability distribution used in the simulation study for the different intention states as a function of the influencing factors. To keep the list short, factors are only included that were of relevance to the scenarios presented in Section 6.3. States marked in bold are used unless otherwise specified. $\mathcal{N}(\mu, \sigma)$ indicates a truncated normal distribution with expected value μ, standard deviation σ, and limited to be larger than 0. For binary states the probability of "true" is given. Discrete states are given in the order specified in Table 6.2.

Intention	Influencing factor	Prior distribution
$\mathcal{I}_{AT}$	Maneuverability: low	$\mathcal{N}(480\,\text{s}, 80\,\text{s})$
	Maneuverability: medium	$\mathcal{N}(360\,\text{s}, 75\,\text{s})$
$\mathcal{I}_{CC}$	Ship type: commercial	0.99
$\mathcal{I}_{CS_\rho}$	None	[0.2, 0.2, 0.2, 0.2, 0.2]
$\mathcal{I}_{GS}$	Ship type: commercial	0.995
$\mathcal{I}_{P_\rho}$	**Relative ship size: similar**	[0.05, 0.90, 0.05]
	Relative ship size: larger	[0.01, 0.59, 0.4]
$\mathcal{I}_{RC}$	Maneuverability: medium, Location: open sea	$\mathcal{N}(1\,\text{km}, 175\,\text{m})$
$\mathcal{I}_{RCF}$	Maneuverability: medium, Location: open sea	$\mathcal{N}(1.5\,\text{km}, 250\,\text{m})$
$\mathcal{I}_{SD}$	**Maneuverability: medium, Location: open sea, Speed: slow**	$\mathcal{N}(300\,\text{m}, 75\,\text{m})$
	Maneuverability: low, Location: open sea, Speed: slow	$\mathcal{N}(700\,\text{m}, 100\,\text{m})$
$\mathcal{I}_{SDF}$	Maneuverability: medium, Location: open sea, Speed: slow	$\mathcal{N}(500\,\text{m}, 120\,\text{m})$
$\mathcal{I}_{SDM}$	Maneuverability: medium, Location: open sea, Speed: slow	$\mathcal{N}(300\,\text{m}, 75\,\text{m})$
$\mathcal{I}_{SS}$	Maneuverability: medium, Location: open sea	$\mathcal{N}(7\,\text{km}, 1.7\,\text{km})$
$\mathcal{I}_{U}$	None	0.9999

Decision criteria

The first approach considers whether the own-ship should consider the reference-ship in the collision avoidance algorithm. Collision avoidance algorithms similar to [135] do not need to consider the reference ship if the own-ship has a *SO* role, and the reference ship is planning to give way. A new node can be introduced into the network to evaluate whether the reference ship is planning to give way or not. A threshold can be proposed that defines how likely it must be that the reference ship will give way for it to be safely ignored by the collision avoidance algorithm.

The node representing whether the reference ship is planning to give way depends on whether the reference ship has a *GW* role (R_ρ), considers it a risky situation (RS_ρ), and if its definitions of ample time ($\mathcal{I}_{AT}$), safe-distance at CPA ($\mathcal{I}_{SD}$), safe distance when crossing in front ($\mathcal{I}_{SDF}$), and safe distance to the current midpoint ($\mathcal{I}_{SDM}$) are acceptable. Additionally, the reference ship is assumed not to give way if it acts in an unmodeled manner ($\mathcal{I}_U$). Equation (6.24) shows the logic statement that defines whether the ship will give way towards ship ρ (WGW_ρ).

$$\begin{aligned} WGW_\rho[t] = & \big(R_\rho == GW\big) \wedge \big(\mathcal{I}_{AT} > AT_{min}\big) \\ & \wedge \big(\mathcal{I}_{SD} > SD_{min}\big) \wedge \big(\mathcal{I}_{SDF} > SDF_{min}\big) \\ & \wedge \big(\mathcal{I}_{SDM} > SDM_{min}\big) \wedge RS_\rho[t] \wedge \neg \mathcal{I}_U \end{aligned} \tag{6.24}$$

Candidate trajectories

The second approach evaluates whether a candidate trajectory for the reference ship is compatible with the estimated intentions. Measurements can be evaluated based on the candidate trajectory and inserted into the network. The network can then be used to evaluate the probability that this trajectory is compatible with the reference ship's intentions ($C[t]$). These candidate trajectories with corresponding probability can be used as scenarios in scenario-based collision avoidance algorithms similar to [71]

Minor alterations are needed to evaluate the measurements based on trajectories. All measurements that consider that the reference ship is keeping its course and speed are instead evaluated using the candidate trajectory of the reference ship while only assuming that all other ships in the encounter will keep their course and speed. The current course ($\mathcal{M}_C$) and speed ($\mathcal{M}_S$) must be evaluated a bit into the candidate trajectory so that the ship has time to execute the potential evasive action. If the situation has not started, then a trajectory keeping the course and speed will be wrongly given a high probability. This is avoided by setting the current distance ($\mathcal{M}_{D_\rho}[t]$) to zero. The time until CPA ($\mathcal{M}_{TCPA_\rho}$) is not relevant for the candidate trajectories as the entire future motion of the ship is considered as known. Instead, this measurement is set to the minimum acceptable time (AT_{min}). An intention to give way at a shorter time than acceptable will evaluate a high probability for trajectories that keep the course and speed. This makes the collision avoidance algorithm take evasive actions if it is likely that the reference ship will give way at an unacceptable short time before CPA. The rest of the measurements can be evaluated as usual.

There are many different ways of generating candidate trajectories. This work generates trajectories based on line-of-sight guidance, as proposed in [14]. These trajectories are generated by simulating a simple ship model that uses a line-of-sight guidance rule to evaluate a reference course that gradually converges towards the nominal path [80]. The nominal path is assumed to go in a straight line going through the position where the ship was first observed, pointing in the same direction as the ship's course at this point. Adding different constant offsets to the reference course generates different trajectories that quickly move away from and then align parallel to the original course. Figures 6.5 and 6.13 shows the resulting trajectories with a constant offset in speed or course. All the trajectories assume that evasive actions are done at the current time-step and not at future time-steps. This assumption can be acceptable for collision avoidance, as it is enough to know if the other ship will give way in time and to what side it will give way.

6.3 Results

This section presents different simulation scenarios that demonstrate the capabilities of the intention model. Scenario 1 to 7 go into detail on specific situations to highlight how the intention model works. The specific scenarios focus on trajectories the ships can take in the future. The probabilities of different candidate trajectories being compatible with the reference ship's intentions ($C[t]$) are presented. Note that probabilities for all trajectory candidates do not need to sum to 1 as there can be multiple trajectory candidates that are compatible with the intentions of the reference ship.

Scenario 8 to 11 show sets of many different similar situations to demonstrate the sensitivity of the intention model to changes in the situation. In these scenarios the focus is on the underlying intentions and whether the ship will give way (WGW_i). How these states develop as the ships approach each other is shown.

The DBN is in each scenario evaluated using the SMILE [60] library for C++. A separate instance of the model is run for all ships in the encounter.

Scenario 1 - Gradual inference

This scenario demonstrates an ability to identify the intentions based on observations. Figure 6.5 shows two ships meeting on a collision course. The situation is a clear crossing situation where, according to COLREGs Rule 15, the blue ship is responsible for giving way while the red should stand on. The model evaluates a 93% chance that the blue ship will give way (WGW) and a 6% chance that the red ship will give way. The blue ship can give way either by reducing its speed or making a starboard turn.

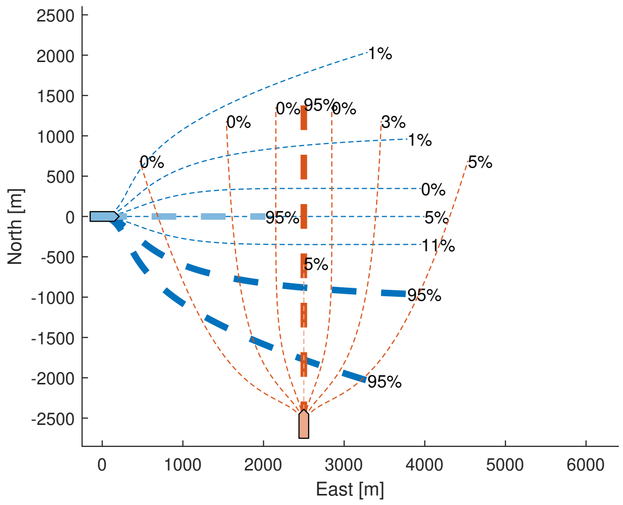


Figure 6.5: Scenario 1. Two ships are meeting on a collision course in a clear crossing situation. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line show the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

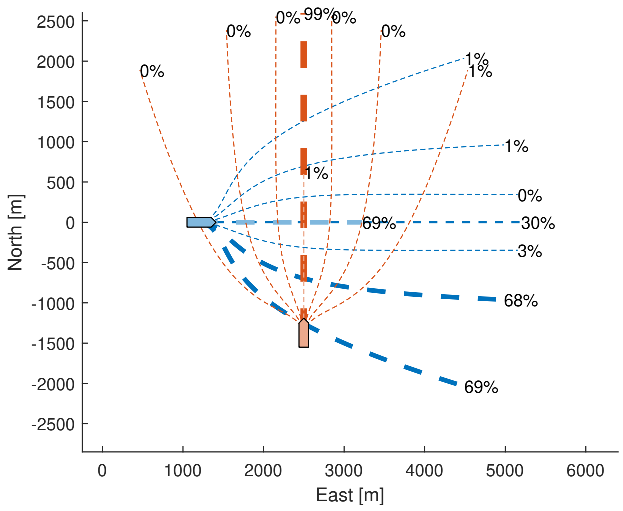


Figure 6.6: Scenario 1. Shows the same encounter as Figure 6.5 at a later time-point. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line show the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

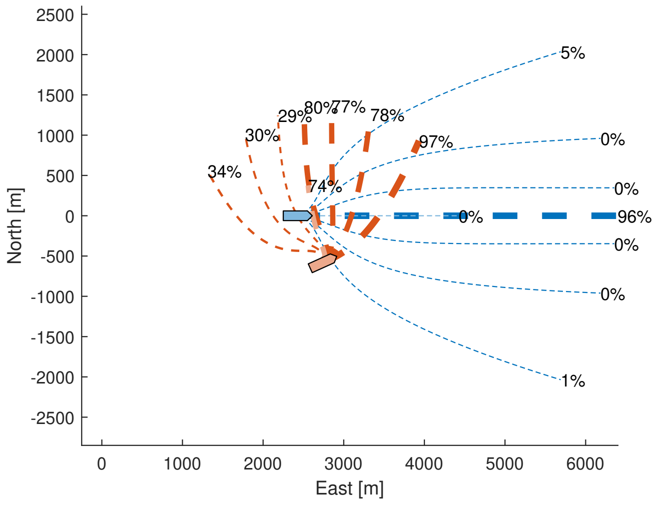


Figure 6.7: Scenario 1. Shows the same encounter as Figures 6.5 and 6.6 at a later time-point. The red ship has changed its course 45°to starboard and halved its speed. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line shows the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

Figure 6.6 shows the same situation at a later time-point. As the blue ship has not yet done any action to avoid collision, it becomes more likely that it believes it has a higher priority making it not give way at all. The model, therefore, evaluates a 68% chance that the blue ship will give way. As the red ship has not changed its course or speed, it becomes less likely that it thinks it has lower priority, which results in a 1% chance that it will give way.

When the red ship starts to make an evasive maneuver, as shown in Figure 6.7, it becomes more likely that the red ship acts to avoid collision. Note that the candidate trajectories are generated relative to the nominal path of the ship, which is assumed to continue northwards. As the time until CPA is very short, it is unlikely that the red ship has such a short definition of ample time. The model, therefore, evaluates a 32% chance that the red ship acts in an unmodeled manner. The probability that the red ship will give way is evaluated to be 29%.

Scenario 2 - COLREGs incompliant action.

This scenario demonstrates the modeling of incompliant behavior. Figure 6.8 shows two ships meeting on a collision course where the blue ship has turned its course to port to cross in front of the red ship. The model predicts that the blue ship will continue to cross in front even though this is not compliant with COLREGs.

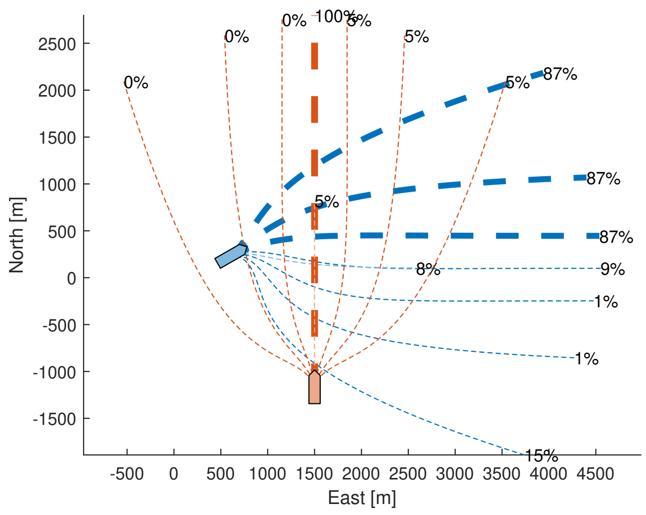


Figure 6.8: Scenario 2. The ships approached in the same manner as shown in Figure 6.5. The blue ship performed a COLREGs incompliant maneuver by changing course to port to avoid collision. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line show the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

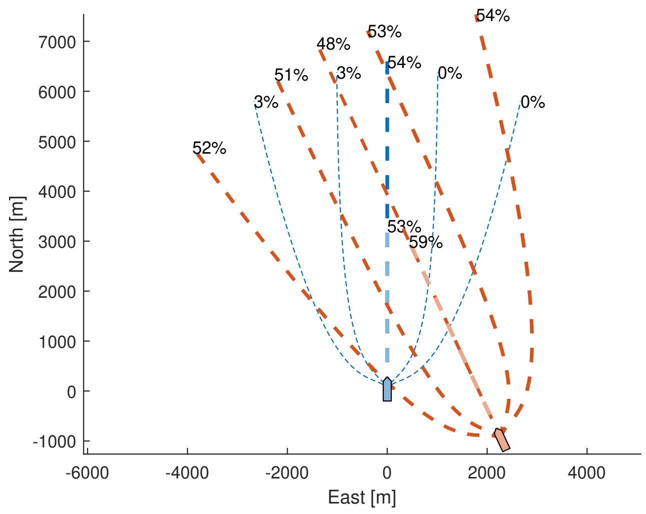


Figure 6.9: Scenario 3. The red ship is approaching the blue ship with a higher speed and a relative bearing of 113 degrees relative to the heading of the blue ship. The bearing is close to the limit between overtaking and crossing, which can cause uncertainty. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line shows the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

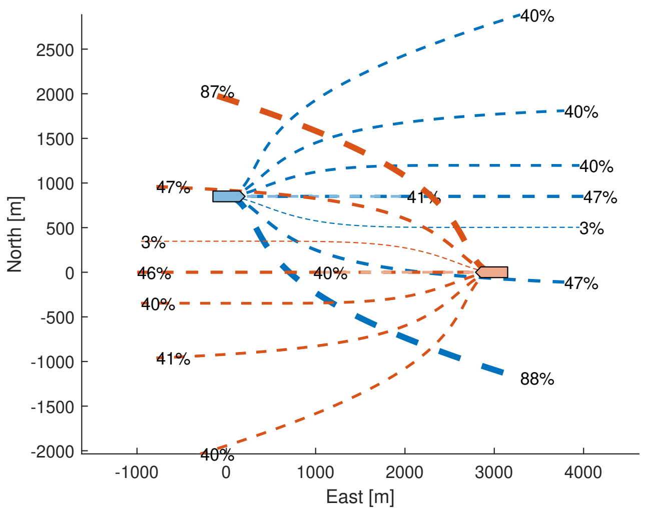


Figure 6.10: Scenario 4. Two ships are approaching in a head-on situation where it is uncertain whether there is a risk of collision (RC_i). The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line show the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

Scenario 3 - Uncertain COLREGs situation

This scenario demonstrates how uncertainty in the COLREGs situation affects the model. Figure 6.9 shows a scenario where the red ship is approaching the blue ship from an angle that is close to the border between an overtaking and a crossing situation. The situation metric evaluates a 54% chance of it being an overtaking situation, in which case the red ship should give way to either side. The remaining 46% is evaluated as a crossing situation, in which case the blue ship should give way behind the red ship. This results in a substantial probability for both keeping the course and speed and taking evasive actions. For the blue ship, none of the candidate trajectories where course alone was changed made the blue ship cross behind the red ship at a safe distance. The only option among the candidate trajectories that gave way behind the red ship was for the blue ship to reduce its speed.

Scenario 4 - Risk of collision

This scenario demonstrates uncertainties that arise from whether there is a risk of collision (RC_i). Figure 6.10 shows two ships meeting in a head-on situation. The model evaluates a 61% chance that there is a risk of collision, and a 86% chance that either ship will give way. If there is no risk of collision, then all actions that keep the ships at a risk-free distance are acceptable. Either way, making a large starboard turn is acceptable as it results in crossing as specified in COLREGs Rule

14.

Scenario 5 - Effect of priors, ship size

Figure 6.11 shows the same scenario as Scenario 1 but utilizes information that the blue ship is significantly larger than the red ship. The model, therefore, evaluates a substantially larger probability that the blue ship has priority over the red, which results in a 58% chance that the blue ship will give way and a 40% chance that the red ship will give way.

Scenario 6 - Effect of priors, maneuverability

Figure 6.12 shows the same scenario as Scenario 1 but with the maneuverability of both ships set to low. This makes it more likely that the blue ship will try to cross with a larger distance between the ships.

Scenario 7 - Multi-ship encounters

Figure 6.13 shows an encounter with three ships, where the red and green ship have a head-on encounter, while the blue ship has an overtaking encounter with the red ship and a head-on encounter with the green ship. If the blue ship had only considered the red ship, then it would be allowed to cross on either side of the ship.

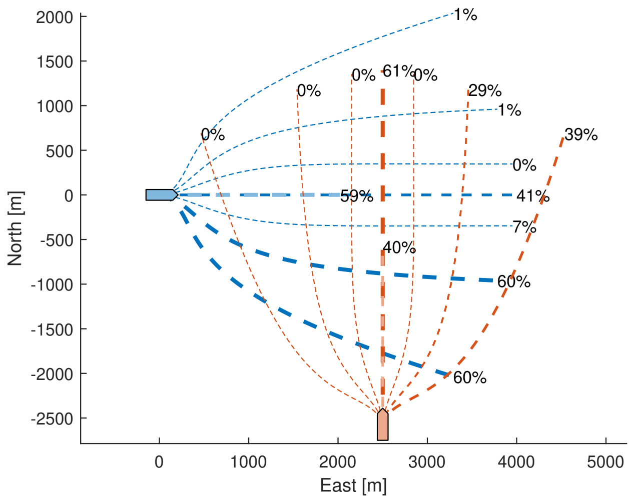


Figure 6.11: Scenario 5. Same situation as Scenario 1. Information that the blue ship is substantially larger than the red ship is inserted as prior information. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line show the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

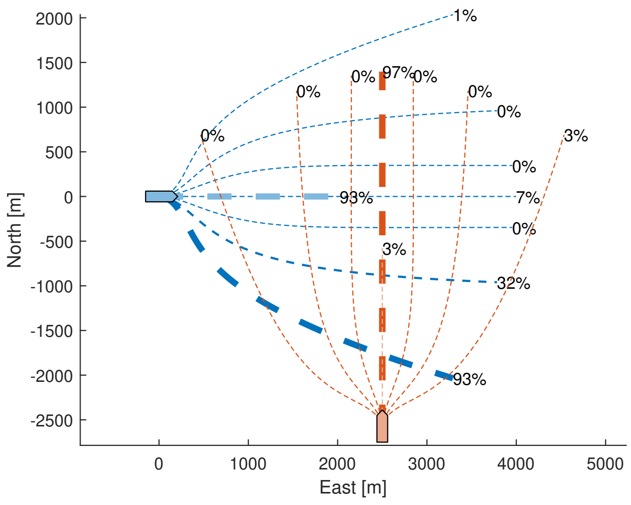


Figure 6.12: Scenario 6. Same situation as Scenario 1. Information that both ships have a low maneuverability is inserted as prior information. The figure shows the different candidate trajectories (dashed lines). The probability at the end of each trajectory and the thickness of the line show the probability that the trajectory is compatible with the ship's intentions ($C[t]$). Trajectories with reduced speed are shown with a lighter color. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

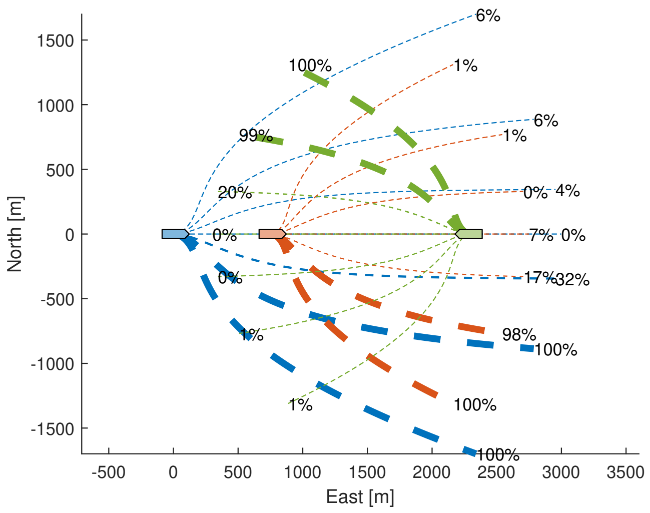
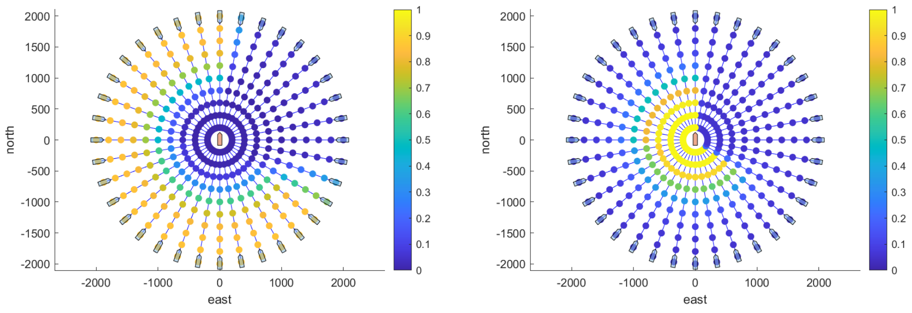


Figure 6.13: Scenario 7. A collision encounter consisting of three ships. The figure shows the different candidate trajectories (dashed lines) with their respective probability of being compatible with the ship's intentions ($C[t]$). The thickness of the line is proportional to the probability of that trajectory being compatible with the ship's intentions. The ship symbols are scaled for visualization purposes and do not represent the true ship size.

As the evasive maneuver has to be correct towards both ships, it can only change its course toward starboard.

Scenario 8 - Entry angle



(a) Will give way (WGW_i) **(b)** Priority higher ($\mathcal{I}_{P_i} = higher$)

Figure 6.14: Multiple different simulations of two ships approaching each other from different angles. The red ship is standing still and the blue ship is keeping its course and speed constant. Each subplot shows how a different state develops as the ships approach. The color shows the probability that the relevant state is “true” at each time step in the different approaches.

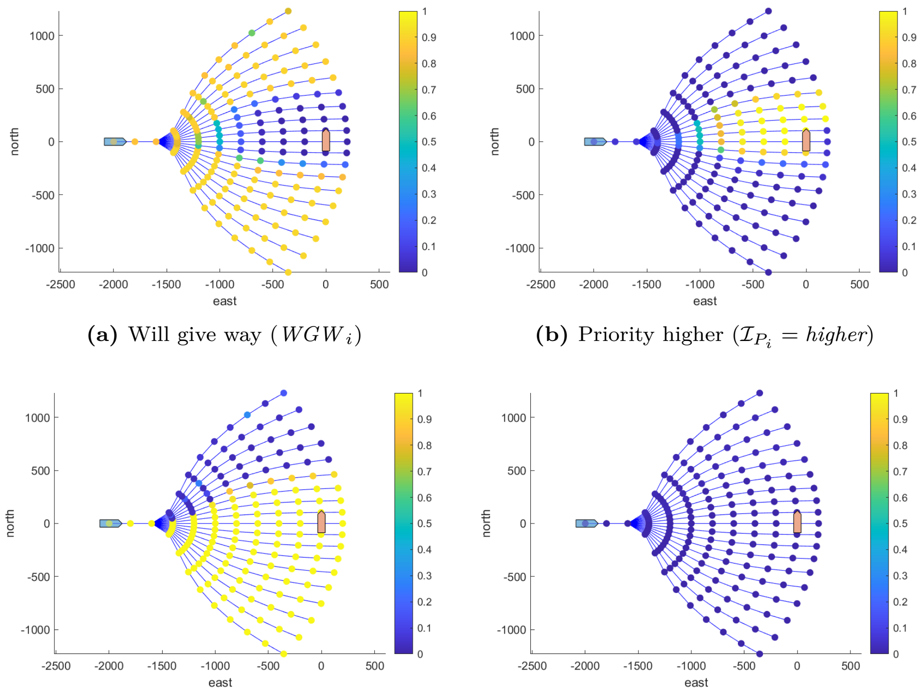
Figure 6.14 shows how the intentions develop as two ships approach from different angles. Figure 6.14(a) shows how the blue ship is initially assumed to give way except if it is approaching in a crossing situation with the red ship on its port side. The gradual transition in whether the blue ship will initially give way demonstrates the gradual transitions between the different COLREGs situations. Figure 6.14(b) shows how the belief that the ship thinks it has higher priority gradually increases as it approaches.

Scenario 9 - Maneuver angle

Figure 6.15 shows how the intentions develop as two ships approach in a crossing situation with different angles on the avoidance action. Figure 6.15(a) shows that as long as the avoidance action is large enough it will be assumed that the blue ship will give way, if not then Figure 6.15(b) shows that it is assumed that the blue ship acts as if it has higher priority. In the cases where the blue ship has changed course to port, Figure 6.15(c) shows a low probability that the blue ship has an intention be COLREGs-compliant when performing evasive maneuvers. In the cases where the port maneuver is small enough to be marked as the ship acting as if it had a higher priority the COLREGs-compliant evasive maneuver will not fall as the model already has an explanation for the observed behavior.

Scenario 10 - Maneuver times

Figure 6.16 shows how the intentions develop as two ships approach in a crossing situation with port and starboard maneuvers at different times. Figure 6.16(a)



(a) Will give way (WGW_i) **(b)** Priority higher ($\mathcal{I}_{P_i} = higher$)

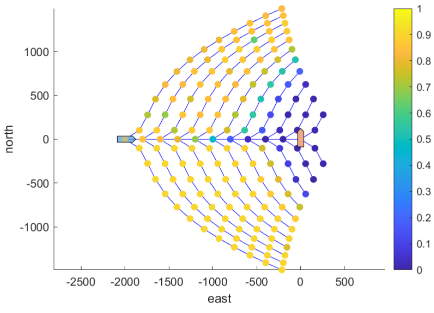
(c) COLREGS-compliant evasive maneuver ($\mathcal{I}_C$) **(d)** Unmodeled behaviour ($\mathcal{I}_U$)

Figure 6.15: Multiple different simulations of two ships approaching in a crossing situation where the blue ship performs an avoidance maneuver with different angles. The red ship is standing still. Each subplot shows how a different state develops as the ships approach. The color shows the probability that the relevant state is "true" at each time step in the different approaches.

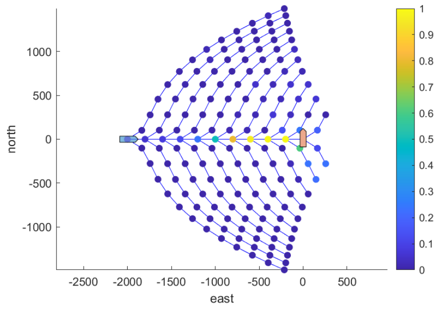
shows that for sufficiently early actions the blue ship will be marked as giving way. In the cases where these actions are to the port side, the COLREGs-compliant evasive maneuver state will drop as shown in Figure 6.16(c). In the cases the actions are too late, Figure 6.16(d) shows that the ship is marked as showing unmodeled behavior as the ship changes course into a collision.

Scenario 11 - Head-on offset

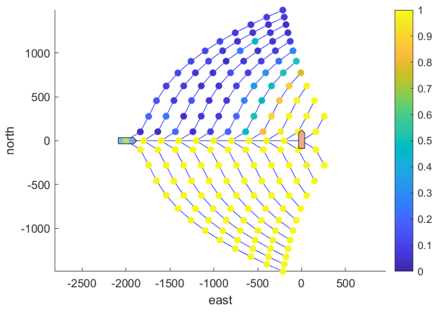
Figure 6.17 shows how the intentions develop as two ships approach in a head-on situation with different sideways offsets. Figure 6.17(a) shows how it was initially assumed that the blue ship will give way as long as the sideways offset is not too large, when it is too large it is assumed that there is no risk of collision as shown in Figure 6.17(d). In the cases where the ships get too close, the behavior is explained with the ship acting as if it has a higher priority, shown in Figure 6.17(b), or that there was no risk of collision, after all, shown in Figure 6.17(d). Figure 6.17(c) does not have any significant changes as the state is only affected by non-compliant evasive maneuvers. As no maneuver is made the behaviour is instead explained



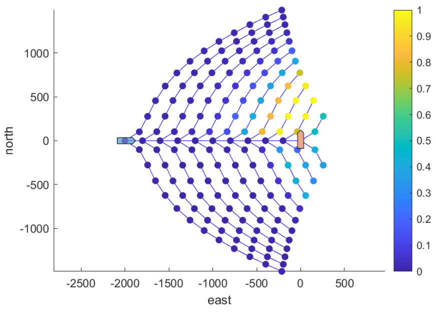
(a) Will give way (WGW_i)



(b) Priority higher ($\mathcal{I}_{P_i} = higher$)



(c) COLREGS-compliant evasive maneuver ($\mathcal{I}_C$)



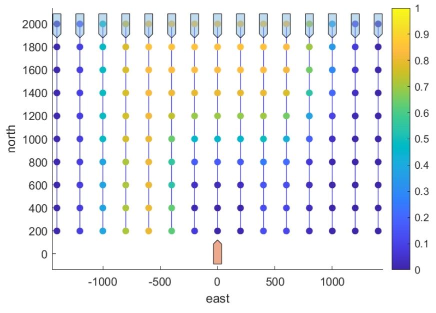
(d) Unmodeled behaviour ($\mathcal{I}_U$)

Figure 6.16: Multiple different simulations of two ships approaching in a crossing situation where the blue ship performs either a starboard or port avoidance action at different times. The red ship is standing still. Each subplot shows how a different state develops as the ships approach. The color shows the probability that the relevant state is "true" at each time step in the different approaches.

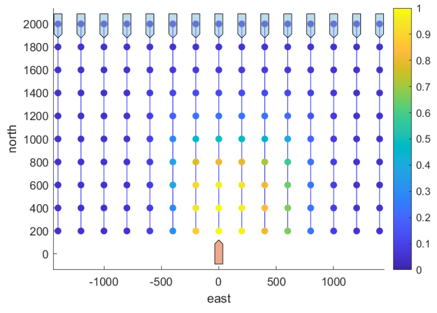
with a higher priority.

6.4 Discussion

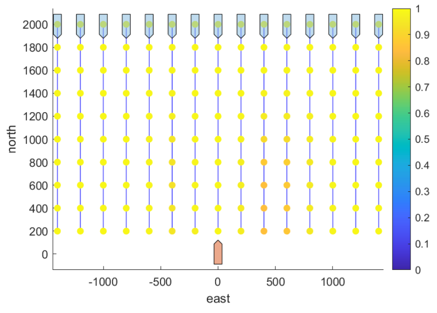
Scenario 1 demonstrates that the model is able to infer the intentions of a ship based on its observed position, course, and speed. The blue ship did not change its course as it approached. This behavior could be explained by the blue ship having high priority or by having a short ample time. Once the ships came closer, the probability that the blue ship had a definition of ample time that was lower than the remaining time until CPA decreased. This increased the probability that the blue ship had higher priority. The red ship changed its course and speed shortly before CPA to avoid collision. Before this point in time, the model did not increase the chance that the red ship would give way as it did not give any indications of giving way. When the red ship finally changed course, the time until CPA was very short, making it quite unlikely that the red ship had such a short definition of ample time. As this behavior does not fit very well with the model, a high chance was evaluated that



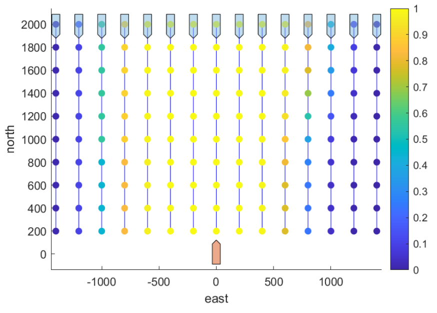
(a) Will give way (WGW_i)



(b) Priority higher ($\mathcal{I}_{P_i} = higher$)



(c) COLREGS-compliant evasive maneuver ($\mathcal{I}_C$)



(d) Risk of collision ($\mathcal{I}_{RC}$)

Figure 6.17: Multiple different simulations of two ships approaching in a head-on situation with different sideways offsets. The red ship is standing still and the blue ship is keeping its course and speed constant. Each subplot shows how a different state develops as the ships approach. The color shows the probability that the relevant state is "true" at each time step in the different approaches.

the red ship acts in an unmodeled way. A collision avoidance algorithm using this intention inference module should display conservative behavior when unmodeled behavior is observed. This will be the case when evaluating candidate trajectories, as all trajectories will have an increased probability of being compatible. When using the intentions as decision criteria, unmodeled behavior will count as not giving way, thereby making the own-ship give way.

1 and 2 show that having multiple different intention variables that can explain a ship's behavior increases the fidelity of the model. In both scenarios, the blue ship acted in a COLREGs incompliant manner. Modeling how the ships are incompliant enables the model to distinguish between Scenario 1 where the blue ship will stand on and the red ship must give way, and Scenario 2 where the blue ship does an evasive maneuver, although to the wrong side.

Modeling of the underlying causes that can cause misunderstandings is demonstrated in Scenario 3. Having a clear distinction between the different COLREGs situations is prone to cause misunderstandings, as it is unlikely that the ships will evaluate borderline situations exactly the same. By modeling this uncertainty, it becomes clear that its insufficient to blindly trust the own-ships interpretation of the situation.

In Scenario 4 the uncertainty stems from whether there is a risk of collision. This scenario gives an example where it is insufficient to consider a single parameter for collision avoidance, such as if the ship will give way. In most other situations, the own-ship must give way if the other ship does not fulfill its obligation. In this situation, the opposite is true; if the other ship fulfills its obligations, then both ships must give way. If the other ship keeps its course, then the own-ship can turn a safe situation into a potentially dangerous one by giving way with a significant starboard maneuver, which is required by COLREGs rule 14.

Scenarios 5 and 6 show that additional information, such as the relative ship size or ship maneuverability, can be used to affect the intention probabilities. Having a collision avoidance algorithm that adapts to the current situation is crucial as ships act in very different manners in different situations, such as open waters and inland waterways. The proposed intention model presented in this chapter is a step towards this ability as it gives the collision avoidance algorithm an understanding of how the other ship will act in the current situation.

Scenario 7 demonstrates that the model can consider encounters with multiple ships. The model considers whether an observed position, course, and speed are compatible with the intention towards all vehicles. The model does not consider that the reference ship has an idea of what the other ships plan to do. This could, for example, be that the blue ship in Figure 6.13 predicts that the red ship will make a starboard turn and therefore chooses to take an even larger starboard turn.

Scenarios 8 to 11 demonstrates the sensitivity of the intention models to how the ships meet and act. Additionally, it shows the effect of the initial distribution of the different intention variables. Scenario 8 shows the effect of the situation classifiers given in Figures 6.2 and 6.3 on the initial probability that the ship will give way. Furthermore, it shows the effect of ample times ($\mathcal{I}_{AT}$) initial distribution on when it becomes likely that the ship has a higher priority. Scenario 9 shows the effect of safe distance ($\mathcal{I}_{SD}$) and safe distance fronts ($\mathcal{I}_{SDF}$) initial distributions on what is considered a valid avoidance action. Scenario 11 shows the effect of safe-distance midpoints ($\mathcal{I}_{SDM}$) initial distribution on how far to the side the blue ship has to pass the red ship for it to be considered as giving way. It also shows the effect of risk of collisions ($\mathcal{I}_{RC}$) initial distribution.

Evaluating different candidate trajectories has some advantages, such as being able to better portray situations such as the one shown in Scenario 4. For the trajectories to realistically portray how the reference ship will act, there must be a candidate trajectory that adequately describes the other ship's trajectory. The candidate trajectory and actual trajectory must be close enough to result in the correct collision avoidance behavior for a collision avoidance algorithm utilizing these intentions. Choosing suitable candidate trajectories is not a trivial task. The ones used in this chapter cannot handle more complicated situations, such as those where the ship is unable to act at the initial time-step but can act at a later one and where the reference ships make more drastic or sequential changes in course or speed.

The probabilities associated with each candidate trajectory do not represent the probability that the reference ship will follow this trajectory. Instead, it represents the probability that this trajectory is something the reference ship would consider acceptable when only considering properties related to COLREGs. If it is known that the ship will follow COLREGs and how it defines the different ambiguities such as ample time and safe distance, then all trajectories that adhere to this definition of the rules will be given a 100% probability of being compatible with the intentions.

This chapter has not considered grounding risk or the COLREGs rules regarding traffic separation schemes (Rule 10), narrow channels (Rule 9), and sailing vessels (Rule 12). Regarding traffic separation schemes and grounding risk, generating candidate trajectories will be more challenging as the trajectories must cover the ship's different options, such as following and leaving the traffic separation scheme correctly. In these situations, it might be necessary to dynamically generate the trajectories based on the current circumstances. An additional challenge arises in narrow channels due to stand-on vessels being allowed to change their course to follow the channel [132].

Furthermore, the model does not explicitly consider measurement uncertainties. This should not be a problem as long as the noise is less than $\mathcal{P}_{CIC}$ and $\mathcal{P}_{CIS}$. If the noise is substantial, then measurement uncertainty should be modeled as well. This can be achieved by having separate nodes representing the measured state and the measurement itself. The measurements themselves should be child nodes of the measured state, and their CPTs should describe the measurement uncertainty. This way of modeling is called the measurement idiom [63].

The model assumes all initial changes in course are large enough to avoid collision without requiring additional course changes. This assumption does not hold if the model is fed an observation in the middle of a course change. The model can then evaluate that the ship is not standing on correct (as it changed its course), nor is it giving way correct (as the course change is too small to avoid collision). This can be handled by introducing a node indicating whether the other ship is currently changing its course.

To have acceptable computational time, the number of time-steps in the DBN must be limited. This can be achieved with a sliding window approach where only the last couple of observations are considered [56]. The priors for the intention nodes must be updated to represent the information that is no longer inside the window. This is done by setting the intention priors equal to what the posterior was at the last time-step that is no longer in the window. With a limited window, the frequency of new observations inserted into the model must be considered. Feeding information more often makes the window consider a shorter time span which will contain more similar observations. This will reduce the inference capabilities of the model. Feeding information less often makes the model respond to changes slower. Not all measurements need to be saved as a time-step in the DBN. The newest time-step of the DBN could be updated at a quick frequency and then only saved as a new time-step if it contained substantial new information relative to the previously saved time-steps. This should make the DBN respond quickly and keep a high inference quality with a limited window.

An alternative to the probabilistic approach presented in this chapter would be to utilize a rule-based system. These rules could for example specify that a ship does not fulfill its give-way obligation if it gets closer to the own-ship than a specified threshold. The probabilistic approach presented in this chapter has advantages compared to such a rule-based system. First, it is able to utilize statistics on how ships have historically acted in similar situations. This makes the proposed method rely less on ad-hoc parameters such as at which distance a ship is no longer considered to fulfill its give-way obligations. Secondly, a probabilistic approach can handle unclear situations by communicating the uncertainty in the intention and trajectory probabilities, and by being able to infer how the other ship most likely interpreted the situation based on the observed behavior. Lastly, the probabilities produced by the intention inference module can enable collision avoidance algorithms to make holistic decisions when multiple pieces of uncertain information has to be considered in light of each other.

6.5 Conclusion

This chapter presents a novel approach for modeling and inferring the intentions of other ships in a potential collision encounter at sea. The simulation study shows that the method is able to infer the state of different intention nodes, identify situations that are likely to lead to misunderstanding, and adapt the intention probabilities to the current situation. This opens up for new possibilities for collision avoidance algorithms. It could enable collision avoidance algorithms to act more safely and predictably as they will better understand the future motion of meeting traffic. They could become able to take early proactive actions to turn a situation prone to misunderstandings into a clear situation where all ships agree on how to act. Lastly, it opens up for collision avoidance algorithms to adapt to the current situation, such as relative ship size and locations. This is an essential feature for collision avoidance algorithms working in multiple different situations where different tuning parameters are needed.

The focus of this chapter is the enhanced modeling and inference capabilities achieved with the proposed framework. Future work is needed on expanding the model to include the parts of COLREGs that were not considered, to consider grounding, to consider factors outside of COLREGs that affect how ships behave, and to validate the model with historical data. Furthermore, work is needed on gathering the statistics that work as priors for the different intention states and on identifying how they are affected by available information on the current situation. Lastly, collision avoidance algorithms must be developed that can utilize the increased situational awareness provided by this model.

Chapter 7

Application of intention model in sea trials

This chapter is based on the following publication

[52] T. Tengesdal, **S. V. Rothmund**, E. A. Basso, T. A. Johansen, and H. Schmidt-Didlaukies, "Obstacle Intention Awareness in Automatic Collision Avoidance: Full Scale Experiments in Confined Waters," *Submitted to Field Robotics*, 2022

The PSB-MPC algorithm was mainly developed by T. Tengesdal, with S. V. Rothmund developing the COLREGs violation cost and interface with the intention model. The intention model is developed by S. V. Rothmund. The main software platform was developed by T. Tengesdal. S. V. Rothmund developed software related to the intention model. Experiments were prepared by T. Tengesdal with help from S. V. Rothmund, E. A. Basso, and H. Schmidt-Didlaukies. Supervision was provided by T. A. Johansen. The first draft was written by T. Tengesdal and S. V. Rothmund. Revision was provided by T. A. Johansen.

7.1 Introduction

7.1.1 Motivation

A key part of achieving autonomy for maritime surface vessels is robust and deliberate collision avoidance (COLAV) systems. These systems are responsible for providing both a safe and efficient avoidance solution in situations where there is a risk of collision with nearby dynamic and static obstacles. To ensure safety, the COLAV system must provide maneuvers that ensure a low risk of collision with static and dynamic obstacles, in addition to adherence to the regulations put forth by the COLREGs [13]. Furthermore, the system should also ensure progress along the intended ship trajectory and strive toward low energy expenditure.

COLAV can be split into two parts. The first part is having adequate situation awareness. situation awareness can be divided into three levels, perception, comprehension, and projection [12]. For COLAV at sea, perception represents identifying the position, speed, and course of the different ships, comprehension represents identifying the intentions of the ships, and projection represents identifying the future motion of the ships. This chapter focuses on the comprehension and projection part of situation awareness by using a modified version of the intention model presented in Chapter 6.

The second part of collision avoidance considers decision-making based on the situation awareness. A challenge when developing decision-making algorithms for collision avoidance is the need to plan COLREGs compliant and optimal trajectories while being able to react quickly when changes are observed. A hybrid approach for handling this challenge is to divide the COLAV planning system into multiple levels, as done in e.g. [135], [142], [143]. In a three-layer structure, as in [135], the highest level is a planner, which runs at low frequency and is responsible for finding a globally optimal trajectory to the ship's goal location while taking static obstacles into account. The mid-level planner is responsible for handling the collision avoidance and compliance with COLREGs in the local area, thus needing to take both static and dynamic obstacles into account. Lastly, the low-level planner is designed to operate at a high frequency in order to handle reactive collision avoidance when new situational information renders the trajectories planned by the higher levels unsafe. In this chapter, the focus is placed on the mid-level COLAV planning, which based on the understanding of the intentions of other ships, finds a collision-free trajectory that complies with the COLREGs regulations [144].

To fully enable both situation awareness and decision-making for collision avoidance, there is also a need for efficient computational platforms, which can both handle and take advantage of the increasing amounts of situational information made available today through modern sensor technology and AIS data. As certain situations can require the COLAV system to consider thousands of possible evasive own-ship maneuvering decisions, static obstacles, and dynamic obstacle intention scenarios with inherent uncertainty, an important part of the decision-making will be to process situation awareness information and possible decision candidates efficiently.

7.1.2 Previous Work

Maritime COLAV has been an active research field since the 1950s [145], and many algorithms have been proposed for solving this problem. This literature review will consider the subset of proposed methods that tackle static and dynamic obstacles, that explicitly consider uncertainties present in dynamic obstacle kinematics, and uncertainty in their intentions, without assuming inter-ship communication. For a general comprehensive literature review on maritime COLAV, please refer to [133], [134], [145]–[148]. For a review on COLAV studies where inter-ship communication is assumed, see [149].

The work in [150] developed an A-star search-based trajectory planner for finding COLREGS-compliant and collision-free trajectories considering dynamic and static obstacles in a lattice. Monte Carlo (MC) simulation with fuzzy logic and trajectory history data was used to find the set of most probable dynamic obstacle trajectories. From the set, the most probable trajectory is considered in the collision avoidance module. No prediction uncertainty was considered for the dynamic obstacles, and the computational efficiency of the planner was only tested in a specific setting.

A* search is applied to collision-free lattice-based trajectory planning [21], where trajectory deviation, collision risk, and non-compliance of COLREGS are penalized in the cost function. An intention-based motion model is used for dynamic obstacles, which uses historical data in order to classify a vessel as COLREGS-compliant or not, and which incorporates reactive COLAV. The details on this model are however not given.

A two-layered COLAV planning system with a global and local lattice-based trajectory planner was developed in 2017 [151]. Here, Voronoi Diagrams together with Fermat‘s Spiral is used to generate a continuous path free of static obstacles. Local re-planning windows are used for taking detected dynamic and static obstacles into account, where the dynamic obstacle motion is predicted under uncertainty using the Constant Velocity (CV) model [152]. However, this can be overly conservative, as the CV model often has unrealistic uncertainty growth in real-world cases [153]. Furthermore, the run-time properties of the local planner were not studied.

Rapidly exploring Random Trees (RRTs) was used in a sampling-based COLAV planning algorithm for COLREGS-compliant dynamic and static collision avoidance in [154]. The planner used a joint simulator for predicting both the own-ship and dynamic obstacle motion, with potential fields being used in the joint prediction to ensure collision-free trajectories. The planning algorithm was shown to be real-time feasible through multiple simulations. However, the joint simulator assumes that nearby ships will always perform deterministic COLREGS-compliant maneuvers if possible, which is not necessarily the case in practice.

Different types of Velocity Obstacle (VO) based methods have been proposed for COLAV in [25], [155]–[158], where the core idea is to compute a set of reachable velocities for the own-ship which do not cause collision with nearby dynamic obstacles. COLREGS adherence and dynamic obstacle kinematic uncertainties have been considered by specifying additional constraints when computing the VO, in probabilistic versions of the algorithm [25], [156]. These methods do however assume constant velocity for dynamic obstacles, unless their trajectories are known beforehand, and can be classified as reactive approaches since the sense-act methodology is used.

In [136], nonlinear VO (NL-VO) is used for the own-ship, together with a method for estimating the intent of vessels having a give-way role. The trajectories of dynamic obstacles with give-way roles are processed using the Douglas-Peucker algorithm, to find their action parameterized as turning points. The reachable velocity of the obstacle at these action points is then checked for intersection with the corresponding own-ship generated NL-VO set, and is used for determining whether the own-ship having stand-on role should perform emergency evasive maneuvers.

Fast probabilistic velocity obstacle (fPVO) for multi-ship COLAV is introduced in [158], where the own-ship roles with respect to all nearby vessels are calculated based on a symmetric own-ship - target ship COLREGS role classification method and used to determine whether or not an evasive maneuver should be taken. The evasive maneuver taken minimizes a cost function that penalizes high collision risk, COLREGS role violation, and trajectory deviation. However, the study assumes that all involved vessels adhere to the COLREGS and try to avoid collision, which is not always valid in practice. The method is extended into a reciprocal (R-fPVO) version in [159], using a DBN for intent inference as in [160]. The Bayesian network estimates the probability of an obstacle ship being COLREGS-compliant or not, and maps this into a rule-compliance factor which scales the COLREGS role violation cost. In contrast, the work presented in this article uses an intention inference method able to infer multiple intention states which in total models the behavior of meeting traffic.

The Scenario-based Model Predictive Control (SB-MPC) for maritime COLAV was first presented in [14] and was extended to the Probabilistic SB-MPC (PSB-MPC) in [144] to explicitly handle dynamic obstacle kinematic uncertainty through the estimation of collision probabilities associated with pairs of predicted own-ship and obstacle trajectories. The method was further extended in [71] to incorporate intent information, where it was shown able to take into account alternative obstacle trajectory scenarios under kinematic uncertainty. The MPC was refined in [20] to consider intent information by using a set of estimated probabilities for a set of uncertain predicted obstacle trajectory scenarios and implemented on a Graphical Processing Unit (GPU) with anti-grounding in [53]. Prior to the present research, the PSB-MPC COLAV planning algorithm had not been tested with an intention inference module, only with usage of a priori intention information.

7.1.3 Proposed Method

Previous work on automatic maritime COLAV has most often neglected or simplified the situational awareness part of COLAV, assuming constant behavior in speed and course for nearby dynamic obstacle ships without uncertainty. This assumption of constant course and speed for vessels involved in encounters will not hold in practice, and limits the own-ship decision-making, possibly leading to more reactive avoidance maneuvers being made. Recently, more studies are emerging that have considered the uncertainty in intent and kinematics of nearby obstacle ships to different extents [21], [133], [150], [151], [159], [161].

Furthermore, an aspect that has not been considered by most COLAV planning algorithms, is the vagueness of the COLREGS and that nearby vessels can have different perspectives and inclinations towards adhering to the rules. Following the rules blindly in any situation can prove to be detrimental [130]. On several points the rules are open to disagreements, making it unsafe to act only based on your own interpretation of a situation [162]. This is also discussed in [163], which illustrates the uncertainties related to the interpretation and adherence of several COLREGS rules. Clearly, inter-ship communication would aid in making safe COLREGS-compliant decisions, but this cannot be relied on being the case in general. Therefore, estimating how and if nearby ships follow the COLREGS under uncertainty will be important in order to resolve hazardous situations safely.

Thus, in this article, we propose a COLAV system with an intention inference module as in [51] for providing added situational information to the PSB-MPC planning algorithm as in [20], [53], and validate the system in experimental trials. All obstacle ships involved in the encounters broadcast their GNSS information for easier tracking system handling, which lets the present article focus on the projection part of situational awareness. The intention information comes in the form of probabilities for likely future obstacle ship trajectories and probabilistic inclinations toward how the ship adheres to the COLREGS. The probability information, predicted trajectories, and other dynamic obstacle information is combined with static obstacle data from navigational charts and fed into the PSB-MPC COLAV planning algorithm. The PSB-MPC utilizes parallel processing to handle larger amounts of situational information and possible own-ship avoidance decisions than what would be possible on a sequential computation platform. This results in increased situational awareness for the COLAV system, when using the uncertain information on the kinematics and intents of nearby dynamic obstacles in the planning. To the authors‘ knowledge, this study represents the first field experimental validation of an intention module for estimating if and how dynamic obstacles adhere to the COLREGS, in a risk-based deliberate COLAV planning algorithm.

The contributions of this article are thus as follows.

- Exploitation of a new intention model for obstacle ships in a risk-based COLAV for increased situational awareness.

- First experimental validation of an intention-aware COLAV system with parallel processing capabilities.

7.1.4 Chapter Overview

The chapter is structured as follows. Section 7.2 gives an overview of the COLAV system architecture and software framework, in addition to the experimental platform used. Sections 7.3 and 7.4 details the intention inference module and the PSB-MPC COLAV planning algorithm. Experimental results are given in Section 7.5 and discussed in Section 7.6 and conclusions are lastly given in Section 7.7.

7.2 System Architecture and Experimental Setup

7.2.1 COLAV System

An overview of the system architecture when using the PSB-MPC COLAV planning algorithm with a situation awareness system including intention inference is given in Figure 7.1. Relevant system components are described below:

GHM Grounding Hazard Manager. Responsible for processing all data related to static obstacles, in a parameterization suitable for the PSB-MPC. A list of relevant hazards, typically inside a radius of $d_{so,relevant}$ around the current own-ship position, is sent to the PSB-MPC. electronic navigational chart (ENC) data is used to get grounding hazard information.

DOM Dynamic Obstacle Manager. Responsible for processing all data related to dynamic obstacles, and for generating prediction scenarios. A prediction scenario here refers to an alternative maneuver or trajectory for the obstacle. The dynamic obstacle data includes tracking system information and estimated intention information for confirmed tracked obstacles. A list of data from relevant dynamic obstacles, inside a radius of $d_{do,relevant}$, is sent to the PSB-MPC. The dynamic obstacle data specifically includes, among others, the current time estimates and error covariances, predicted trajectories, and estimated probabilities for each alternative maneuvering scenario.

DOII Dynamic obstacle intention inference. Updates the intention model based on observed behavior. Receives dynamic obstacle data with predicted trajectories for each obstacle, and evaluates the probability that the obstacle will follow different candidate trajectories. See Section 7.4.

PSBMPC Finds the optimal own-ship trajectory based on static and dynamic obstacle data. See Section 7.3.

7.2.2 The Own-ship Platform

The field experiments used the fully actuated milliAmpere 2 vessel as the own-ship platform, shown in Figure 7.2. The vessel is owned by NTNU and is used for research and technology development purposes in the field of autonomous maritime transport in urban areas. It is 8.6 m long, 3.5 m wide, fully electric, and equipped with four Fischer Panda 10 kW azimuth thrusters. To obtain accurate own-ship navigation data, the ferry uses a moving base real time kinematics (RTK) solution with two global navigation satellite system (GNSS) receivers and two RTK antennas. At the time of the experiments, a virtual reference station (VRS) was used with a Network Transport of RTCM data over IP (NTRIP) client for corrections, due to an RTK base station being down.

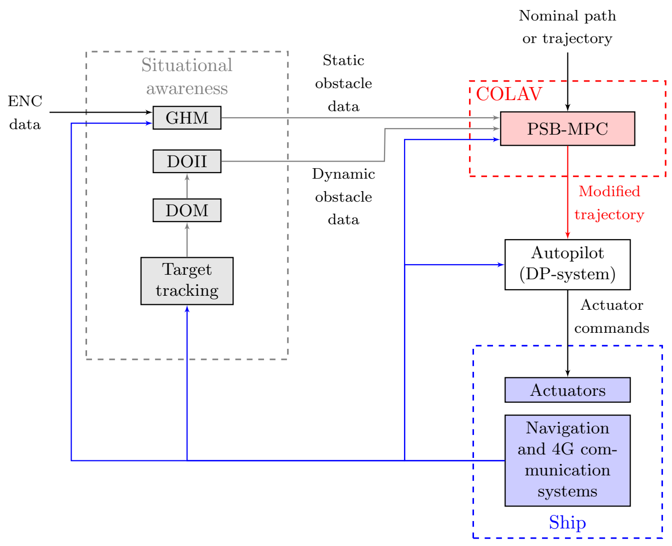


Figure 7.1: COLAV system overview used in the experiments.

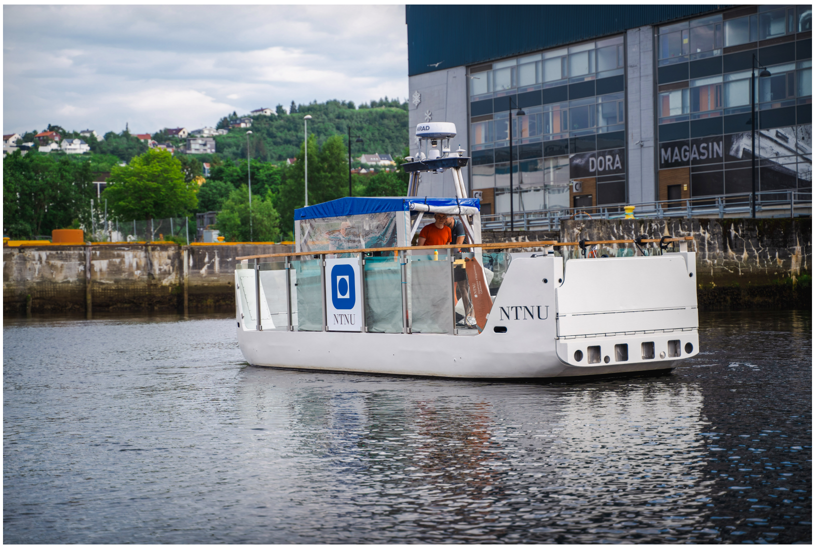


Figure 7.2: The Milliampere 2 ferry used as the autonomous ship in the experiments. Courtesy of Mikael Sætereid.

The robotic operating system (ROS) is used as middleware for the COLAV system, with software packages for each system component. The Milliampere 2 ferry uses a commercial dynamic positioning (DP) system from Marine Technologies, which takes in trajectory references in planar pose, velocity, and acceleration. It is configured to perform small heading changes, mostly relying on speed changes, since back-and-forth ferry transportation is the goal. This created some challenges for trajectory tracking and COLREGs adherence in this work, as it is important to make apparent maneuvers to comply with COLREGs rule 8.

The PSB-MPC COLAV planning algorithm calculates the desired trajectory for the ferry to follow, and converts the output to the correct format of input for the DP system. The PSB-MPC and the situation awareness module runs on a separate computer, which is connected to an onboard Milliampere 2 computer via ethernet. The onboard computer is responsible for enabling autonomy on the ferry and connects to the navigation, sensor, and DP systems. The separate computer is a workstation running with Ubuntu 20.04.3 LTS as its operating system with an Intel(R) Core(TM) i9-10900K 3.70 GHz processor, 32 GB RAM, and an NVIDIA GeForce RTX 3090 GPU.

7.2.3 Target Tracking

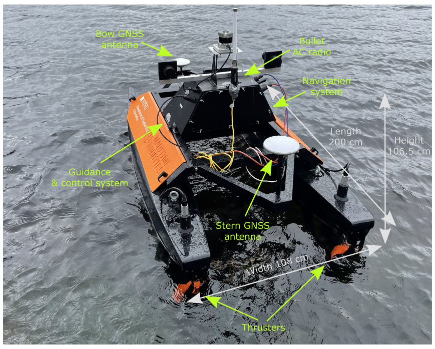
As the main research objective is to showcase how an intention inference module can be used with a COLAV planning algorithm for safer and more efficient ship guidance, we use a simple communication setup where dynamic obstacles send GNSS information to the own-ship tracking system. The tracking system then filters these measurements using a linear KF [66] to produce tracks for each vessel, using a constant velocity (CV) model [164] for the estimation, as done in simulation in [144]. We then use 4G routers on each vessel to establish communication.

The target ships (dynamic obstacles (DOs)) considered in the experiments is a Jeanneau Marlin 65 vessel called Havfruen depicted in Figure 7.3(a), and the Cyberotter [1] depicted in Figure 7.3(b). To get GNSS information from Havfruen, the vessel-driver runs a laptop with a Ublox ZED-F9P receiver that sends position measurements over ROS to the own-ship using ROS Ublox driver software. The Cyberotter has an onboard SBG GNSS system, which in the same fashion sends its GNSS information over ROS. The own-ship is here configured as the ROS master, running on the workstation on milliAmpere 2, which is connected via ethernet and the 4G communication link to Havfruen and the Cyberotter.

Due to the Milliampere 2 ferry being restricted to operate within channels and harbor areas without waves, all experiments were performed in the easternmost basin of Nyhavna, Trondheim, Norway.

[1] More information at https://otter.itk.ntnu.no/doku.php?id=cyberotter.

(a) Havfruen. Courtesy of Mannhullet at NTNU.



(b) Cyberotter [165].

Figure 7.3: The target ships or dynamic obstacles Havfruen and Cyberotter used in the experiments.

7.3 The PSB-MPC COLAV Planning Algorithm

The PSB-MPC [20], [53] is a finite control set MPC [166], [167], and uses sampling-based optimization for deliberate COLAV planning. It considers a finite set of own-ship trajectories and finds the optimal one which minimizes a cost function

$$\mathcal{H}^l(t_0) = \mathcal{H}^l_{do} + \mathcal{H}^l_{colregs} + \mathcal{H}^l_{so} + \mathcal{H}^l_p \tag{7.1}$$

at the current time t_0. The cost function penalizes dynamic obstacle collision risk cost $\mathcal{H}^l_{do}$, breaching the COLREGs with respect to all dynamic obstacles $\mathcal{H}^l_{colregs}$, grounding on static obstacles $\mathcal{H}^l_{so}$, in addition to penalizing deviations from the nominal trajectory $\mathcal{H}^l_p$, respectively. Here, l is the index of the control behavior or own-ship trajectory being considered. The control behavior consists of n_M sequential avoidance maneuvers taken by the own-ship. It is parameterized by a sequence $\left[(U^l_{m,1}, \chi^l_{m,1}), ..., (U^l_{m,n_M}, \chi^l_{m,n_M})\right]$ consisting of speed multiplicative factors U^l_m and additive course angle offsets χ^l_m, which are applied to the autopilot references U_d and χ_d in speed over ground (SOG) and course over ground (COG) at different time steps in the prediction horizon. The optimal control behavior is found as

$$l^*(t_0) = \arg\min_l \mathcal{H}^l(t_0) \tag{7.2}$$

where the solution gives us an optimal avoidance trajectory for the own-ship, which is regarded as a reference trajectory to be tracked by the DP-system. A sampling time of 5 seconds is used for the MPC. The PSB-MPC problem is illustrated in Figure 7.4. Note that the control behaviors displayed here are for illustrative purposes.

Details on the prediction models for the own-ship and the DOs, how the grounding cost ($\mathcal{H}^l_{so}$) and path deviation costs ($\mathcal{H}^l_p$) are calculated, details on how the algo-

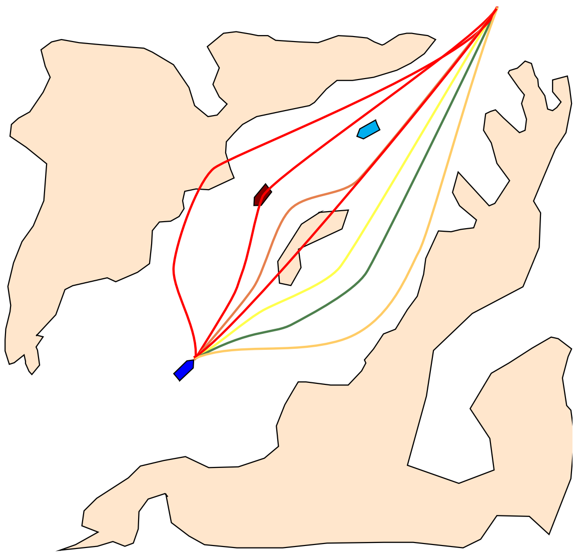


Figure 7.4: PSBMPC illustration, with the own-ship running the algorithm in blue. Nearby dynamic obstacles are shown in cyan and brown, and grounding hazards in beige. Candidate control behaviors predicted by the MPC are shown, where the color from red to green represents the cost, with green being the lowest. Thus, the green candidate trajectory is the optimal one. The figure is from [53]

.

rithm can be parallelized for running on a graphical processing unit (GPU), and how autopilot references are constructed based on the desired trajectory are not included in this thesis. They can be found in the article this chapter is based on [52].

7.3.1 Cost Function

Dynamic Obstacle Cost

The dynamic obstacle cost captures the probabilistic collision risk and is here chosen as a sum over all nearby obstacles

$$\mathcal{H}^{l}_{do} = \sum_{i=1}^{n_{do}} \mathcal{H}^{l,i}_{do} \tag{7.3}$$

where the individual dynamic obstacle cost is given by a weighted sum over the n^i_{ps} prediction scenarios

$$\mathcal{H}^{l,i}_{do} = \sum_{s=1}^{n^i_{ps}} \hat{\mathbb{P}}^i_s \mathcal{H}^{l,i,s}_{do} \tag{7.4}$$

with the prediction scenario probabilities $\{\hat{\mathbb{P}}^i_s\}_{s=1}^{n^i_{ps}}$ as weights. The prediction scenario probabilities are evaluated with the dynamic obstacle intent inference (DOII) module. Details on the collision cost associated with the pair of an own-ship control behavior l and a dynamic obstacle i behaving as in prediction scenario s ($\mathcal{H}^{l,i,s}_{do}$) can be found in [52].

COLREGS Violation Cost

The COLREGs violation cost has here been updated, relative to [20], [53], to consider the entire candidate trajectory of the own-ship when evaluating whether or not the rules were breached and is meant to make adherence to the COLREGs rules $8, 13, 14, 15, 16$ and 17 easier. The total COLREGs cost, $\mathcal{H}^{l}_{colregs}$ is evaluated by summing the COLREGs violation costs over all dynamic obstacles

$$\mathcal{H}^{l}_{colregs} = \sum_{i=1}^{n_{do}} \mathcal{H}^{l,i}_{colregs} \tag{7.5}$$

which enables multi-ship COLREGs adherence. The cost towards a specific dynamic obstacle i for a control behavior l is evaluated by summing the violation costs for all different prediction scenarios s weighted by the prediction scenario probabilities as

$$\mathcal{H}^{l,i}_{colregs} = \sum_{s=1}^{n^{i}_{ps}} \hat{\mathbb{P}}^{i}_{s} \mathcal{H}^{l,i,s}_{colregs} \tag{7.6}$$

where the scenario-specific cost is

$$\mathcal{H}^{l,i,s}_{colregs} = \kappa_{SO} \mu^{l,i}_{SO} \hat{\mathbb{P}}^{i}_{WGW} + \kappa_{GW} \mu^{l,i}_{GW} \hat{\mathbb{P}}^{i}_{CCEM} + \kappa_{RA} \mu^{l}_{RA} \tag{7.7}$$

and $\mu^{l,i,s}_{SO}$ and $\mu^{l,i,s}_{GW}$ are binary indicators of whether or not the own-ship following control behavior l violated its stand-on or give-way role with respect to dynamic obstacle i behaving as in scenario s. The binary indicator μ^{l}_{RA} is equal to 1 if the control behavior l induces an initial avoidance maneuver that does not lead to a readily apparent action. The cost (7.7) considers the entire own-ship and dynamic obstacle trajectories in the violation evaluation, which is different from previous research [14] where only instantaneous states were considered. Distinct penalty parameters κ_{SO}, κ_{GW}, and κ_{RA} are used to weight the stand-on, give-way, and readily apparent violation costs separately. The stand-on violation cost is weighted by the estimated probability $\hat{\mathbb{P}}^{i}_{WGW}$ that the obstacle will fulfill its give-way obligations when specified by the COLREGs, obtained from the intention inference module. The give-way violation cost is similarly weighted by the estimated probability $\hat{\mathbb{P}}^{i}_{CCEM}$ that the obstacle will perform a COLREGs compliant evasive maneuver when specified by COLREGs, also obtained from the intention inference module. See Section 7.4 for more information.

The COLREGs situation CS is defined when the distance d_{0i} from the own-ship to obstacle ship i at the current time is less than a given parameter $d_{colregs}$. This ensures that subsequent changes in course or bearing do not change the situation, as stated in COLREGs rule 13(d). The COLREGs situation is determined as follows

- If the bearing from the own-ship to the obstacle is more than 22.5° abaft the beam of the own-ship, then the own-ship is being overtaken and $CS = OT\text{-}en$ (COLREGs rule 13(b)).
- If the bearing from the obstacle to the own-ship is more than 22.5° abaft the obstacle beam then the own-ship is overtaking and $CS = OT\text{-}ing$ (COLREGs rule 13(b)).
- If the relative heading between the ships is within $180° \pm 10°$, then there is a head-on situation and $CS = SO$ (COLREGs rule 14).
- If there is no head-on or overtaking situation then there is a crossing situation (COLREGs rule 15), and either a port side or starboard side crossing based on the following:
  - If the bearing to the obstacle ship relative to the own-ship heading is negative, the obstacle ship is on the port side and $CS = CR\text{-}PS$.
  - If the bearing to the obstacle ship relative to the own-ship heading is positive, the obstacle ship is on the starboard side, and $CS = CR\text{-}SS$.

For a control behavior l, no penalty is given if the time until CPA with respect to obstacle i, $t^{l,i}_{cpa}$, is longer than a time threshold T_{start}. Also, no penalty is given if the ships will pass at a distance at CPA, $d^{l,i}_{cpa}$, that is larger than a distance threshold $d_{colregs}$, i.e. $d^{i}_{cpa} > d_{colregs}$. Note that all distance calculations take the extent of the own-ship and an obstacle into account.

A control behavior l gets a stand-on violation penalty if the own-ship has a stand-on role and the control behavior leads to a change in course (CIC^l) of more than $CIC^{SO}_{max} = 15°$, or change in speed (CIS^l) of more than $CIS^{SO}_{max} = 0.5\,\text{m/s}$. No stand-on penalty is given if the current distance d_{0i} to the obstacle is below a critical value $d_{critical} = 15\,\text{m}$, as all ships should then act to avoid collision (COLREGs rule 17 (a)(ii) and (b)).

$$\begin{aligned}\mu^{l,i}_{SO} =& t^{l,i}_{cpa} < T_{start} \wedge d^{l,i}_{cpa} < d_{colregs} \wedge d_{0i} > d_{critical} \quad \wedge \\ & (CS == OT\text{-}en \vee CS == CR\text{-}PS) \wedge (CIC^l \neq none \vee CIS^l \neq none)\end{aligned} \tag{7.8}$$

A control behavior l gets a give-way violation penalty if it does not act as specified in COLREGs rules 14, 15, and 17. COLREGs rule 14 specifies that ships in a head-on situation should pass port-to-port ($P2P$). According to COLREGs rule 15 a ship in a crossing situation with the other on its starboard side $CR\text{-}SS$, should cross aft of that ship (CA). If a ship is forced to take action in a crossing situation with the other on its port side, $CR\text{-}PS$, then it should avoid changing the course to port (COLREGs rule 17(c)).

$$\begin{aligned}\mu^{l,i}_{GW} =& t^{l,i}_{cpa} < T_{start} \wedge d^{l,i}_{cpa} < d_{colregs} \wedge (CS == SO \wedge \neg P2P^l \quad \vee \\ & CS == CR\text{-}SS \wedge \neg CA^l \vee CS == CR\text{-}PS \wedge CIC^l == port)\end{aligned} \tag{7.9}$$

Table 7.1: The initial distribution used for the intention variables. $\mathcal{N}(\mu, \sigma)_{[a,b]}$ indicates a normal distribution with expected value μ, standard deviation σ, truncated to be between a and b, and discretized into 30 evenly spaced intervals. The probability of "true" is given for binary states.

Symbol	Prior
$\mathcal{I}_{AT}$	$\mathcal{N}(60\,\mathrm{s}, 4\,\mathrm{s})_{[0,100]}$
$\mathcal{I}_{CC}$	0.98
$\mathcal{I}_{CS_\rho}$	$\mathcal{M}_{CS_\rho}$
$\mathcal{I}_{GS}$	0.99
$\mathcal{I}_{P_\rho}$	$[\mathit{higher} = 0.05, \mathit{similar} = 0.90, \mathit{lower} = 0.05]$
$\mathcal{I}_{SD}$	$\mathcal{N}(25\,\mathrm{m}, 2.5\,\mathrm{m})_{[0,30]}$
$\mathcal{I}_{SDF}$	$\mathcal{N}(20\,\mathrm{m}, 4\,\mathrm{m})_{[0,50]}$
$\mathcal{I}_{SDM}$	$\mathcal{N}(15\,\mathrm{m}, 2.5\,\mathrm{m})_{[0,30]}$
$\mathcal{I}_{U}$	0.00001

Lastly, a control behavior l gets a readily apparent violation penalty if it does not adhere to COLREGs rule 8, which will be the case if the behavior induces any non-zero initial speed or course modification that is not sufficiently high to make the resulting avoidance maneuver apparent. Here, any non-zero course modification which is less than $CIC^{GW}_{min} = 45°$ or any non-zero speed modification less than $CIS^{GW}_{min} = 0.5\,\mathrm{m/s}$ gains a violation:

$$\mu^{l,i}_{RA} = t^{l,i}_{cpa} < T_{start} \wedge d^{l,i}_{cpa} < d_{colregs} \quad \wedge \tag{7.10}$$

$$((|\chi^{l}_{m,1}| > 0 \wedge |\chi^{l}_{m,1}| < CIC^{GW}_{min}) \vee (U^{l}_{m,1} > 0 \wedge U^{l}_{m,1} < CIS^{GW}_{min})) \tag{7.11}$$

The large course change threshold was chosen due to the Milliampere 2 ferry not being tuned to make large heading changes when tracking trajectories.

7.4 Dynamic Obstacle Intention Inference (DOII)

The DOII module gets a list of dynamic obstacle estimates and prediction scenarios from the dynamic obstacle manager (DOM). The goal of the DOII is then to evaluate the probabilities that the dynamic obstacle will follow the different scenarios ($\hat{\mathbb{P}}^i_s$). Additionally, the module gives the probability that the obstacle will fulfill its give-way obligations ($\hat{\mathbb{P}}^i_{WGW}$) and the probability that it will perform a COLREGs compliant evasive maneuver ($\hat{\mathbb{P}}^i_{CCEM}$) when supposed to.

The DOII uses a modified version of the intention model presented in Chapter 6. Changes that are made to the model are presented in Section 7.4.1 and the updated model is shown in Figure 7.5. Section 7.4.2 presents how the computational burden is limited, Section 7.4.3 how the probability of the prediction scenarios are evaluated, and Section 7.4.4 how the probabilities directly used in the COLREGs violation cost are evaluated. Definition of terms is given in Tables 6.2 to 6.4 with new terms in Table 7.2. The prior distributions used in the experiments are found in Table 7.1.

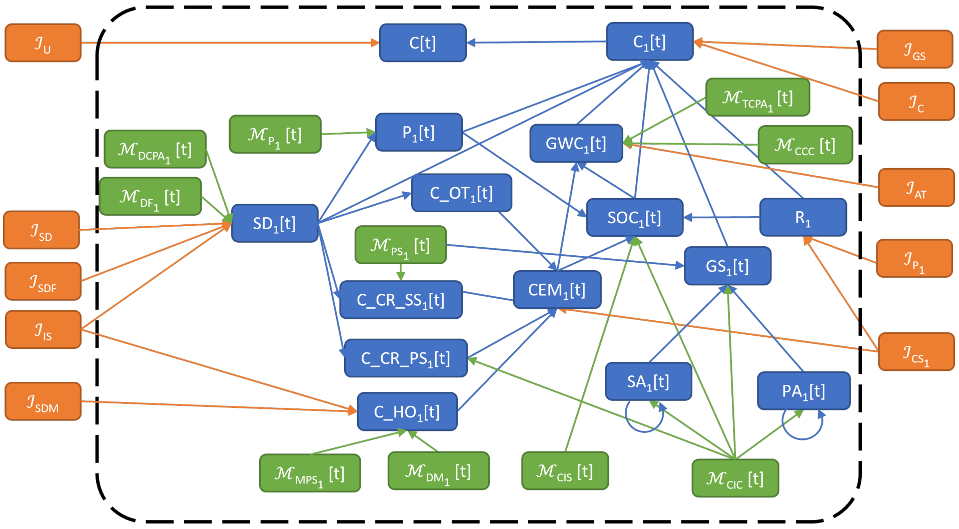


Figure 7.5: Topology of the DBN used in this chapter. The figure shows the case of an encounter with two ships. Everything inside the dotted line is time-varying and repeats for each time step in the DBN. The intention nodes, shown in orange, are time-invariant. The abbreviations are explained in Tables 6.2 to 6.4 and 7.2.

Table 7.2: New measurement variables.

Symbol	Description	States
$\mathcal{M}_{PS_\rho}[t]$	Whether the reference ship will pass with ship ρ on its port or starboard side	$\{starboard, port\}$
$\mathcal{M}_{CCC}[t]$	Whether the reference ship is currently changing course. Takes the state *true* if there has been a course change more than $\mathcal{P}_{CIC} = 10°$ the last $\mathcal{P}_{CCT} = 6\,\mathrm{s}$.	binary

7.4.1 Changes in the intention model

Due to problems with having a constant speed of the DOs during the start of the experiments all changes in speed were ignored in the intention model. The DOs only intentionally changed their course, never the speed, in all of the experiments presented in this chapter.

The largest change done in this chapter is to remove nodes related to situation started (SS) and risky situation (RS). These were removed as they were deemed unnecessary for the experiments presented in this chapter as they are done in confined waters where there is no uncertainty in whether there is a risk of collision and when the situation starts. This change was to simplify the model for experimentation. This change was implemented by setting $SS_\rho = true$ and $RS_\rho = true$ with a 100% probability, and by removing the risk of collision (RC_ρ) node.

As it is assumed known when the situation starts in the experiments, the change in course can be directly measured ($\mathcal{M}_{CIC}$) by subtracting the current course and speed from the initial course and speed. The change in course (CIC) and change in speed (CIS) is therefore set equal to the measured value, while initial course (IC) was removed.

$$CIC = \mathcal{M}_{CIC} \tag{7.12}$$

Similarly, as there is no uncertainty in when the situation starts there is no need for a separate COLREGs situation measurement ($\mathcal{M}_{CS_\rho}$) node. The measurement is instead inserted as the prior distribution of the COLREGs situation intention node ($\mathcal{I}_{CS_\rho}$) as show in Table 7.1.

A limitation in the work presented in Chapter 6 was that it did not handle measurements made during a course change. This is solved in this chapter by introducing a new measurement of whether the ship is currently changing course ($\mathcal{M}_{CCC}$). This measurement considers the course change over the last $\mathcal{P}_{CCT}$ seconds. Choosing a larger $\mathcal{P}_{CCT}$ makes the model identify the reference ship's actions slower. Choosing $\mathcal{P}_{CCT}$ small makes it more likely that a measurement in the middle of a course change is interpreted as an intentional choice of course. 6 s was found to work well in the setting considered in this chapter. If the reference ship is currently changing course then the reference ship will be marked as giving way correctly (GWC_ρ). The updated function for GWC_ρ is then:

$$GWC_\rho[t] = CEM_\rho[t] \lor \big((\mathcal{M}_{TCPA_\rho}[t] > \mathcal{I}_{AT}) \land SOC_\rho[t]\big) \lor \mathcal{M}_{CCC} \tag{7.13}$$

During testing, problems were observed where the model too often went for unmodeled behavior, even when the behavior could be modeled with not being COLREGs compliant or not displaying good seamanship. These were solved by changing where the terms related to good seamanship and COLREGs compliant were included. $\mathcal{I}_{GS}$ and $\mathcal{I}_{CC}$ were removed from the definition of correct evasive maneuver (CEM_ρ) and were instead included in the definition of compatible towards ship ρ (C_ρ) as follows:

$$\begin{aligned} C_\rho[t] = & P_\rho[t] \lor (R_\rho == SO \land SOC_\rho[t]) \\ & \lor \Big(\big((R_\rho == GW \land GWC_\rho[t]) \lor (\neg\mathcal{I}_{CC} \land SD_\rho)\big) \land (GS_\rho \lor \neg\mathcal{I}_{GS})\Big) \end{aligned} \tag{7.14}$$

An error in Equation (6.7) for multi-ship encounters when one of the ships has passed the reference ship is solved by updating the stand on correct (SOC_ρ) equation to include information on whether the ship had passed (P).

$$\begin{aligned} SOC_\rho[t] = &(\mathcal{M}_{CIC}[t] == \mathit{straight} \wedge \mathcal{M}_{CIS}[t] == \mathit{none}) \\ &\vee \left(\exists_{\lambda \in P \setminus \{\rho\}} R_\lambda == GW \wedge CEM_\lambda[t] \wedge \neg P_\lambda[t]\right) \end{aligned} \tag{7.15}$$

During testing a bug in the code which evaluates whether the reference ship passes front or aft of ship p was observed. Due to time limitations, a workaround was proposed instead of solving the problem. The bug was circumvented by redefining the definition of crossing starboard-side evasive maneuver $(C\_CR\_SS_\rho)$ to consider whether it's passing with ship ρ on the port side instead.

$$C\_CR\_SS_\rho[t] = (\mathcal{M}_{PS_\rho} == \mathit{port}) \wedge SD_\rho[t] \tag{7.16}$$

Good seamanship (GS_ρ) was extended to require that if the reference ship changed its course to port then it has to pass with ship ρ on its starboard side, and vice versa.

$$GS_\rho[t] = \neg(SA_\rho[t] \wedge PA_\rho[t]) \wedge \mathcal{M}_{CIC}[t] \neq \mathcal{M}_{PS_\rho}[t] \tag{7.17}$$

7.4.2 Limiting computational burden

As the computational burden of evaluating the DBN increases with each new time-steps, the number of time steps has to be limited. This was achieved by in most cases inserting a new measurement on the current time step thereby overriding the previous measurement. A new time-step is made if the previous time a new time-step was made is more than $\Delta_{ts,max} = 20\,\text{s}$, or if the previous time-step was made no less than $\Delta_{ts,min} = 10\,\text{s}$ and either ship in the encounter has changed their course more than $\Theta = 15°$. These bounds were chosen quite large to ensure that the computational time at all times during the experiments would be low enough to allow real-time inference.

7.4.3 Probability of prediction scenarios

The procedure for evaluating trajectory candidates presented in Section 6.2.5 is used. As a large number of prediction scenarios can be compatible with the intentions, the resulting distribution over the different scenarios must be normalized such that it sums to 1. Additionally a prior is introduced to bias the distribution towards trajectory candidates that keep the reference ship's current course. The prior $1 + \cos(\delta_{s0})$ is used, where δ_{s0} is the difference between the course held $10\,\text{s}$ into a scenario s and the current course of the reference ship. The time interval of $10\,\text{s}$ is chosen as it gives enough time for the reference ship to change its course. The constant 1 is used to ensure that the prior is always positive. The probability of a candidate trajectory can then be evaluated as follows where η is a normalization factor:

$$\hat{\mathbb{P}}_s^i = \eta \Pr\{C[t]\}(1 + \cos(\delta_{s0})) \tag{7.18}$$

7.4.4 Direct intention information

The probability $\hat{\mathbb{P}}^i_{WGW}$ considers if the reference ship (obstacle i in the PSB-MPC formulation) will fulfill its give-way obligations, which is dependent on whether it does not have higher priority ($\mathcal{I}_{P_\rho}$), it intends to adhere to COLREGs when performing evasive maneuvers ($\mathcal{I}_{CC}$) and does not display unmodelled behavior ($\mathcal{I}_U$):

$$\hat{\mathbb{P}}^i_{WGW} = \Pr\{(\mathcal{I}_{P_\rho} \neq \mathit{higher}) \wedge \mathcal{I}_{CC} \wedge \neg \mathcal{I}_U\} \tag{7.19}$$

The second probability considers if the reference ship (obstacle i in the PSB-MPC formulation) will perform a COLREGs compliant evasive maneuver, and is given by:

$$\hat{\mathbb{P}}^i_{CCEM} = \Pr\{\mathcal{I}_{CC}\} \tag{7.20}$$

7.5 Experimental Results

The COLAV system described in Sections 7.2 to 7.4 was tested in the following nine different scenarios, with the relevant COLREGs rules indicated in parentheses:

1. Head-on scenario with correct dynamic obstacle behavior, where the obstacle makes a starboard turn (COLREGs rule 14).
2. Head-on scenario with wrong dynamic obstacle behavior, where the obstacle makes a port turn (COLREGs rule 14).
3. Crossing with a dynamic obstacle as stand-on and own-ship as the give-way vessel (COLREGs rule 15 and 16).
4. Crossing with the own-ship as stand-on, and a dynamic obstacle which does not adhere to COLREGs and does not give way (COLREGs rule 15 and 17).
5. Crossing with the own-ship as stand-on, with a COLREGs compliant dynamic obstacle taking a starboard turn (COLREGs rule 15 and 17).
6. Overtaking scenario with the obstacle being overtaken (COLREGs rule 13 and 16).
7. Overtaking scenario with the own-ship being overtaken (COLREGs rule 13 and 17).
8. Combined overtaking and crossing starboard side scenario, with the own-ship overtaking an obstacle and being the give-way vessel for another obstacle (COLREGs rules 13, 15 and 16-17).
9. Combined crossing starboard side and port side scenario (COLREGs rules 15 and 16-17).

Furthermore, COLREGs rule 7 on adequate collision risk assessment and rule 8 on performing apparent actions in ample time are also relevant for all the scenarios.

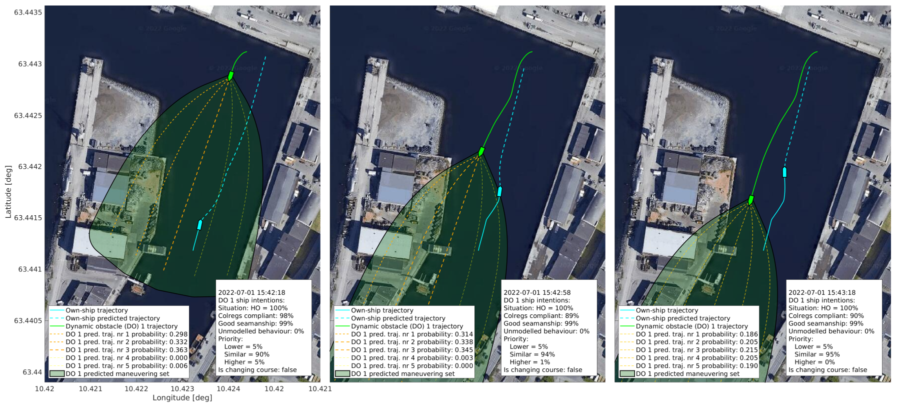
For each scenario, the chosen trajectory of the Milliampere 2 ferry together with the different prediction scenarios is shown at three different time instants (see Figures 7.7 to 7.15). The thickness of the dashed lines, representing the different prediction scenarios, are scaled based on their likelihood as evaluated by the DOII. Additionally, how the intention states develop through the scenarios are shown together with the course and speed of all vessels involved. The priority state indicates whether the vessel will act as if it has a higher priority (ignoring its obligations to give way) or lower priority (ignoring its obligation to stand on). The COLREGs compliant state indicates whether the vessel will ignore all the specifications in COLREGs regarding how to give way, but still try to avoid collision. Good seamanship indicates whether or not the ship shows how it is going to act. Unmodelled behavior indicates all other non-compliant behavior.

In the two-ship scenarios, Havfruen was used as the dynamic obstacle $i = 1$ (DO1). In the three-ship scenarios, the Cyberotter is the second dynamic obstacle $i = 2$ (DO 2). Milliampere 2 was set to track a desired speed of 1 m/s in all scenarios, except when it was overtaking another vessel, and in the combined overtaking and port side crossing scenario, where reference speeds of 2.0 m/s and 1.5 m/s were used, respectively. Low speeds were used due to the experiments being performed in confined waters, and because Milliampere 2 has a max speed limitation of 2.5 m/s. The dynamic obstacles speeds vary from 0.5 m/s to 3.0 m/s.

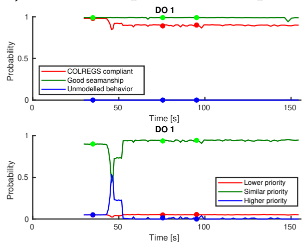
The PSB-MPC COLAV planning algorithm was tuned to reflect the confined space Milliampere 2 was to operate in. Two sequential maneuvers ($n_m = 2$) was considered in the horizon of the MPC, where the second maneuver is taken after $t_{ts} = 60$ s. Thus, the own-ship is planned to perform an initial maneuver at t_0, and a corrective one at $t_0 + t_{ts}$, which does not necessarily return the ship to its nominal trajectory. The own-ship safety zone was set to $d_{safe} = 8.6$ m. The algorithm parameters for grounding cost were chosen to allow the own-ship to maneuver within a margin of approximately 5 m to nearby grounding hazards.

Videos of the experimental work can be found at `https://youtu.be/9cmDqQDgBDc` or by scanning 7.6. Note that the video was shot on only one of the experiment days. Some of the scenarios in the video do therefore not correspond exactly to the scenarios presented in this chapter.

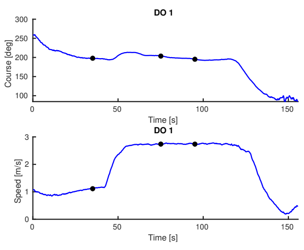
Figure 7.6: Video of experiments. `https://youtu.be/9cmDqQDgBDc`


Latitude [deg]
Longitude [deg]
2022-07-01 15:42:18
DO 1 ship intentions:
Situation: HO = 100%
Colregs compliant: 98%
Good seamanship: 99%
Unmodelled behaviour: 0%
Priority:
Lower = 5%
Similar = 90%
Higher = 5%
Is changing course: false
Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
DO 1 pred. traj. nr 1 probability: 0.298
DO 1 pred. traj. nr 2 probability: 0.332
DO 1 pred. traj. nr 3 probability: 0.363
DO 1 pred. traj. nr 4 probability: 0.000
DO 1 pred. traj. nr 5 probability: 0.006
DO 1 predicted maneuvering set
2022-07-01 15:42:58
Colregs compliant: 89%
Similar = 94%
Higher = 1%
DO 1 pred. traj. nr 1 probability: 0.314
DO 1 pred. traj. nr 2 probability: 0.338
DO 1 pred. traj. nr 3 probability: 0.345
DO 1 pred. traj. nr 4 probability: 0.003
DO 1 pred. traj. nr 5 probability: 0.000
2022-07-01 15:43:18
Colregs compliant: 90%
Similar = 95%
Higher = 0%
DO 1 pred. traj. nr 1 probability: 0.186
DO 1 pred. traj. nr 2 probability: 0.205
DO 1 pred. traj. nr 3 probability: 0.215
DO 1 pred. traj. nr 4 probability: 0.205
DO 1 pred. traj. nr 5 probability: 0.190

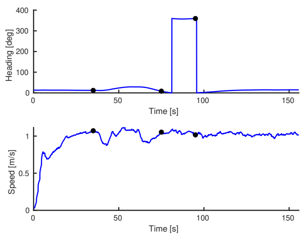
(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.



(b) Dynamic obstacle intention states.



(c) Dynamic obstacle course and speed.



(d) Own-ship heading and speed.

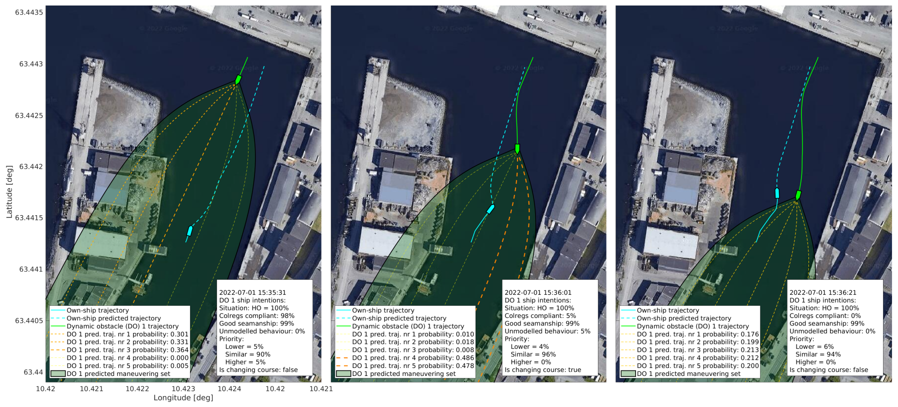
Figure 7.7: Scenario 1 - Head-on with a compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

Scenario 1 - Head-on With a Compliant Dynamic Obstacle

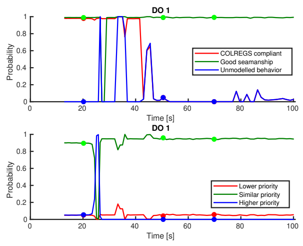
Results from the compliant head-on scenario are given in Figure 7.7 and show that the own-ship performs a COLREGs compliant evasive maneuver. Shortly after the 40 s mark, the intention model starts to infer that the obstacle ship acts as if it has a higher priority than the own-ship. This is due to the ships getting quite close without the dynamic obstacle taking action. Once the dynamic obstacle takes an evasive action, the probability of it having higher priority quickly drops to 0.

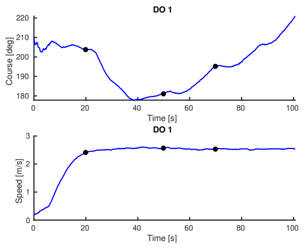
Scenario 2 - Head-on With a Non-compliant Dynamic Obstacle

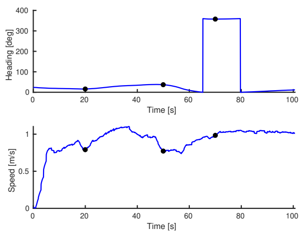
Results from the non-compliant head-on scenario is given in Figure 7.8. The scenario shows that the own-ship starts to perform a COLREGs-compliant evasive maneuver. Similar to the previous scenario, the probability of the dynamic obstacle acting as if it has higher priority increases for a short time as the dynamic obstacle acts quite late. Once the dynamic obstacle changes course to port, the intention model switches between the dynamic obstacle either not being COLREGs compliant or showing unmodelled behavior. At $t = 50$ s it has concluded on the dynamic obstacle not being COLREGs compliant. The knowledge of the obstacle being non-compliant enables the own ship to disregard COLREGs as well and avoid


Latitude [deg]
Longitude [deg]
Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
DO 1 pred. traj. nr 1 probability: 0.301
DO 1 pred. traj. nr 2 probability: 0.331
DO 1 pred. traj. nr 3 probability: 0.364
DO 1 pred. traj. nr 4 probability: 0.000
DO 1 pred. traj. nr 5 probability: 0.005
DO 1 predicted maneuvering set
2022-07-01 15:35:31
DO 1 ship intentions:
Situation: HO = 100%
Colregs compliant: 98%
Good seamanship: 99%
Unmodelled behaviour: 0%
Priority:
Lower = 5%
Similar = 90%
Higher = 5%
Is changing course: false
DO 1 pred. traj. nr 1 probability: 0.010
DO 1 pred. traj. nr 2 probability: 0.018
DO 1 pred. traj. nr 3 probability: 0.008
DO 1 pred. traj. nr 4 probability: 0.486
DO 1 pred. traj. nr 5 probability: 0.478
2022-07-01 15:36:01
DO 1 ship intentions:
Situation: HO = 100%
Colregs compliant: 5%
Good seamanship: 99%
Unmodelled behaviour: 5%
Priority:
Lower = 4%
Similar = 96%
Higher = 0%
Is changing course: true
DO 1 pred. traj. nr 1 probability: 0.176
DO 1 pred. traj. nr 2 probability: 0.199
DO 1 pred. traj. nr 3 probability: 0.213
DO 1 pred. traj. nr 4 probability: 0.212
DO 1 pred. traj. nr 5 probability: 0.200
2022-07-01 15:36:21
DO 1 ship intentions:
Situation: HO = 100%
Colregs compliant: 0%
Good seamanship: 99%
Unmodelled behaviour: 0%
Priority:
Lower = 6%
Similar = 94%
Higher = 0%
Is changing course: false

(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.







(b) Dynamic obstacle intention states.

(c) Dynamic obstacle course and speed.

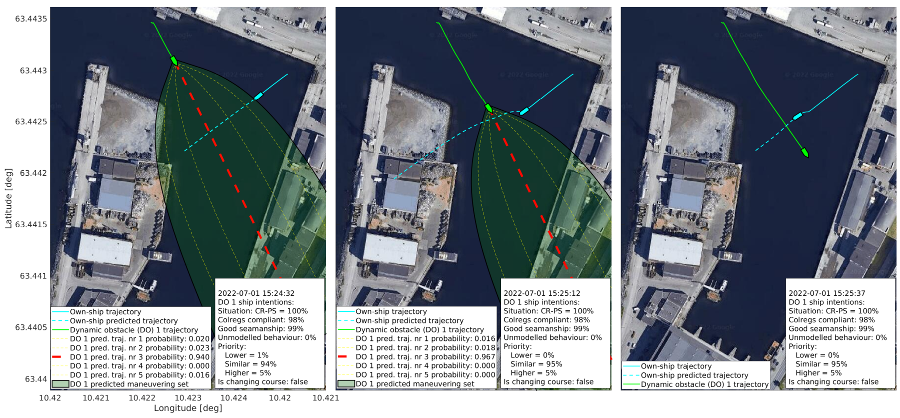
(d) Own-ship heading and speed.

Figure 7.8: Scenario 2 - Head-on with a non-compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

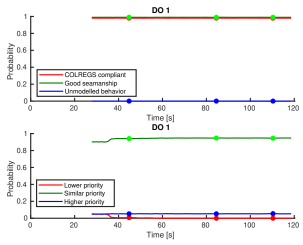
collision through a port avoidance maneuver.

Scenario 3 - Crossing With the Own-ship as Give-way Vessel, and a Compliant Stand-on Dynamic Obstacle

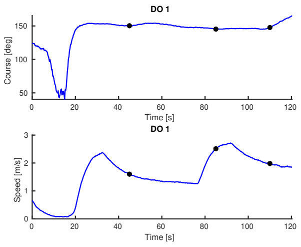
Figure 7.9 shows results for the starboard side crossing. The own-ship performs a COLREGs-compliant evasive maneuver by changing its course to starboard and reducing its speed to avoid collision. Reducing the speed is the most effective action in this case as the confined spaces make larger course changes more susceptible to grounding. Furthermore, the Milliampere 2 thruster configuration and DP-system are configured for small heading changes and slow movements, which makes it easier to change speed than to alter the course. The collision avoidance algorithm is tuned such that Milliampere 2 should still try to change its course as this can be easier to see from the other vessel's point of view.


2022-07-01 15:24:32
DO 1 ship intentions:
Situation: CR-PS = 100%
Colregs compliant: 98%
Good seamanship: 99%
Unmodelled behaviour: 0%
Priority:
Lower = 1%
Similar = 94%
Higher = 5%
Is changing course: false
Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
DO 1 pred. traj. nr 1 probability: 0.020
DO 1 pred. traj. nr 2 probability: 0.023
DO 1 pred. traj. nr 3 probability: 0.940
DO 1 pred. traj. nr 4 probability: 0.000
DO 1 pred. traj. nr 5 probability: 0.016
DO 1 predicted maneuvering set
2022-07-01 15:25:12
DO 1 pred. traj. nr 1 probability: 0.016
DO 1 pred. traj. nr 2 probability: 0.018
DO 1 pred. traj. nr 3 probability: 0.967
DO 1 pred. traj. nr 4 probability: 0.000
DO 1 pred. traj. nr 5 probability: 0.000
Lower = 0%
Similar = 95%
Higher = 5%
2022-07-01 15:25:37
Latitude [deg]
Longitude [deg]

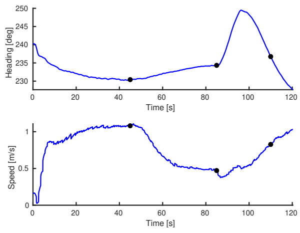
(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.



(b) Dynamic obstacle intention states.



(c) Dynamic obstacle course and speed.



(d) Own-ship heading and speed.

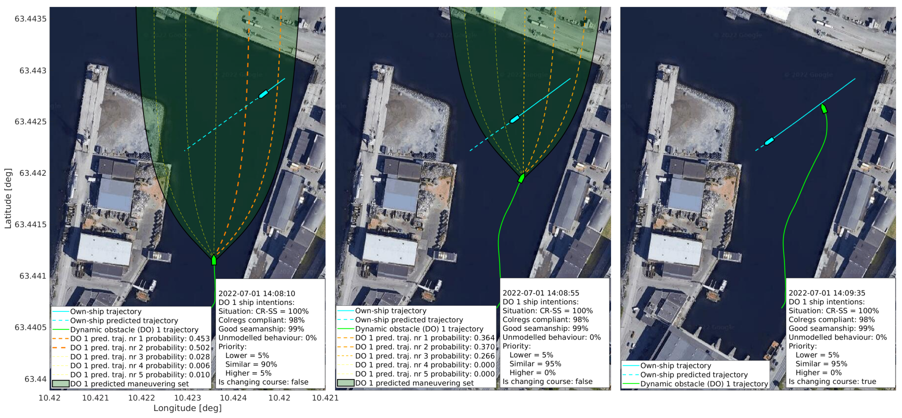
Figure 7.9: Scenario 3 - Crossing with the own-ship as give-way vessel, and a compliant stand-on dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

Scenario 4 - Crossing With the Own-ship as Stand-on Vessel, and a Compliant Dynamic Obstacle

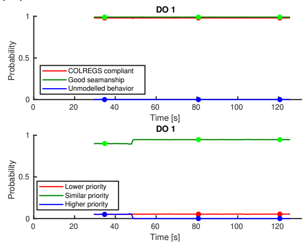
Results from this scenario are given in Figure 7.10. The intention model correctly predicts that the obstacle ship will make a starboard maneuver to avoid collision. The own-ship can therefore keep its course and speed without increasing the risk of collision. This fulfills the requirements that stand-on vessels shall keep their course and speed (unless forced to give way), as specified in COLREGs rule 17.

Scenario 5 - Crossing With the Own-ship as Stand-on and a Non-compliant Dynamic Obstacle

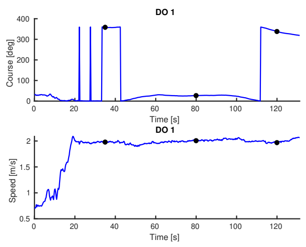
Results from the port side crossing scenario with a non-compliant dynamic obstacle are given in Figure 7.11. The probability of the dynamic obstacle acting as if it has a higher priority gradually increases as the obstacle ship comes closer without significantly changing its course or speed. Once it is quite likely that the dynamic obstacle will not give way, the own-ship decides to half its speed to avoid a potential collision. This shows that the resulting algorithm is able to deviate from the stand-


Latitude [deg]
Longitude [deg]
2022-07-01 14:08:10
DO 1 ship intentions:
Situation: CR-SS = 100%
Colregs compliant: 98%
Good seamanship: 99%
Unmodelled behaviour: 0%
Priority:
Lower = 5%
Similar = 90%
Higher = 5%
Is changing course: false
Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
DO 1 pred. traj. nr 1 probability: 0.453
DO 1 pred. traj. nr 2 probability: 0.502
DO 1 pred. traj. nr 3 probability: 0.028
DO 1 pred. traj. nr 4 probability: 0.006
DO 1 pred. traj. nr 5 probability: 0.010
DO 1 predicted maneuvering set
2022-07-01 14:08:55
Similar = 95%
Higher = 0%
DO 1 pred. traj. nr 1 probability: 0.364
DO 1 pred. traj. nr 2 probability: 0.370
DO 1 pred. traj. nr 3 probability: 0.266
DO 1 pred. traj. nr 4 probability: 0.000
DO 1 pred. traj. nr 5 probability: 0.000
2022-07-01 14:09:35
Is changing course: true

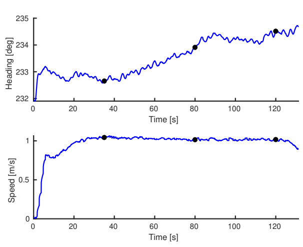
(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.



(b) Dynamic obstacle intention states.



(c) Dynamic obstacle course and speed.



(d) Own-ship heading and speed.

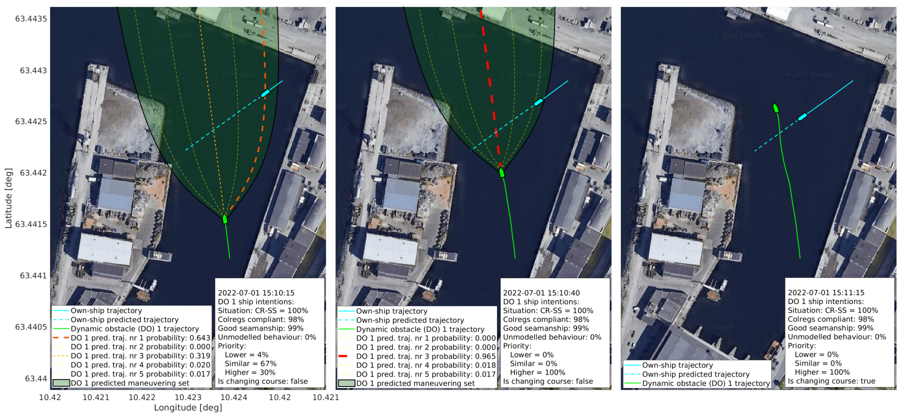
Figure 7.10: Scenario 4 - Crossing with the own-ship as stand-on vessel, and a compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

on requirements when needed, as specified in COLREGs rule 17(b). The algorithm also adheres to rule 17(c) by not changing its course to port.

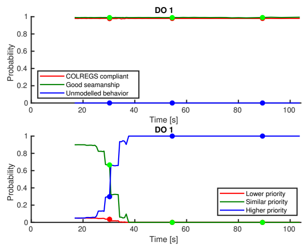
Scenario 6 - Overtaking

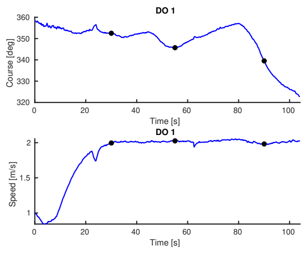
Results from the overtaking scenario are given in Figure 7.12. The scenario shows that the own ship is able to avoid collision while overtaking, but the intention predictions are not ideal. This is due to the intention module interpreting all course changes larger than some threshold as being done with an intention. This does not work well in this scenario as the dynamic obstacle is holding a too low velocity for keeping its course steady enough, as can be seen in Figure 7.12(c). This experiment could not be performed at a higher speed due to the speed limitations of Milliampere 2.

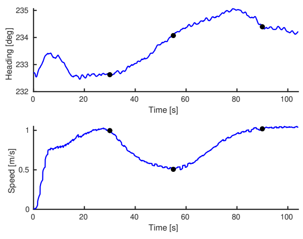
The intention module concludes at the 40 s mark that the dynamic obstacle is displaying unmodelled behavior. This state can explain all possible behaviors and thus gives all future scenarios equal likelihood. The variations in the rest of the states after the 40 s mark can therefore not be caused by new observations as the



(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.







(b) Dynamic obstacle intention states. **(c)** Dynamic obstacle course and speed. **(d)** Own-ship heading and speed.

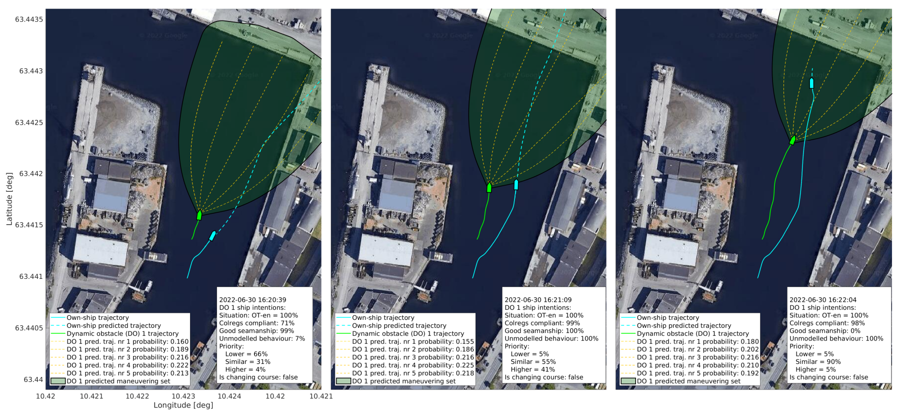
Figure 7.11: Scenario 5 - Crossing with the own-ship as stand-on and a non-compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

observations are already explained by the unmodelled behavior state, nor do they affect the likelihood of different scenarios. We believe that these variations are computational quirks caused by the states being unobservable.

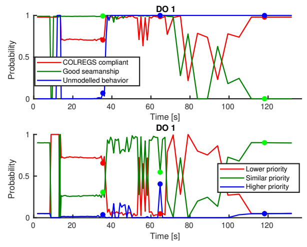
Scenario 7 - Overtaken

The COLAV system is shown to also handle its stand-on role when being overtaken, with results given in Figure 7.13. A bit after the $t = 60\,\text{s}$ mark the dynamic obstacle changes course toward the own-ship. This is to avoid collision with floating platforms not shown in the figure. As the intention model observed that the dynamic obstacle changes course towards what seems like a collision course with the own-ship, it marks the dynamic obstacle as showing unmodelled behavior. The good seamanship decreases right before it is marked as unmodelled behavior as the dynamic obstacle changes course to port while still planning to cross with the own-ship on its port side. Similar variations in intention states as discussed in Scenario 6 are observed once unmodelled behavior becomes equal to 1.

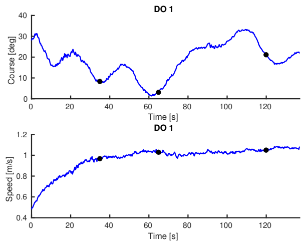
The scenario also shows a weakness in the current PSB-MPC dynamic obstacle pre-


Latitude [deg]
Longitude [deg]
Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
DO 1 pred. traj. nr 1 probability: 0.160
DO 1 pred. traj. nr 2 probability: 0.189
DO 1 pred. traj. nr 3 probability: 0.216
DO 1 pred. traj. nr 4 probability: 0.222
DO 1 pred. traj. nr 5 probability: 0.213
DO 1 predicted maneuvering set
2022-06-30 16:20:39
DO 1 ship intentions:
Situation: OT-en = 100%
Colregs compliant: 71%
Good seamanship: 99%
Unmodelled behaviour: 7%
Priority:
Lower = 66%
Similar = 31%
Higher = 4%
Is changing course: false
DO 1 pred. traj. nr 1 probability: 0.155
DO 1 pred. traj. nr 2 probability: 0.186
DO 1 pred. traj. nr 3 probability: 0.216
DO 1 pred. traj. nr 4 probability: 0.225
DO 1 pred. traj. nr 5 probability: 0.218
2022-06-30 16:21:09
DO 1 ship intentions:
Situation: OT-en = 100%
Colregs compliant: 99%
Good seamanship: 100%
Unmodelled behaviour: 100%
Priority:
Lower = 5%
Similar = 55%
Higher = 41%
Is changing course: false
DO 1 pred. traj. nr 1 probability: 0.180
DO 1 pred. traj. nr 2 probability: 0.202
DO 1 pred. traj. nr 3 probability: 0.216
DO 1 pred. traj. nr 4 probability: 0.210
DO 1 pred. traj. nr 5 probability: 0.192
2022-06-30 16:22:04
DO 1 ship intentions:
Situation: OT-en = 100%
Colregs compliant: 98%
Good seamanship: 0%
Unmodelled behaviour: 100%
Priority:
Lower = 5%
Similar = 90%
Higher = 5%
Is changing course: false

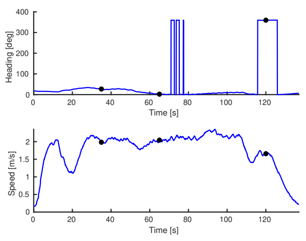
(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.



(b) Dynamic obstacle intention states.



(c) Dynamic obstacle course and speed.



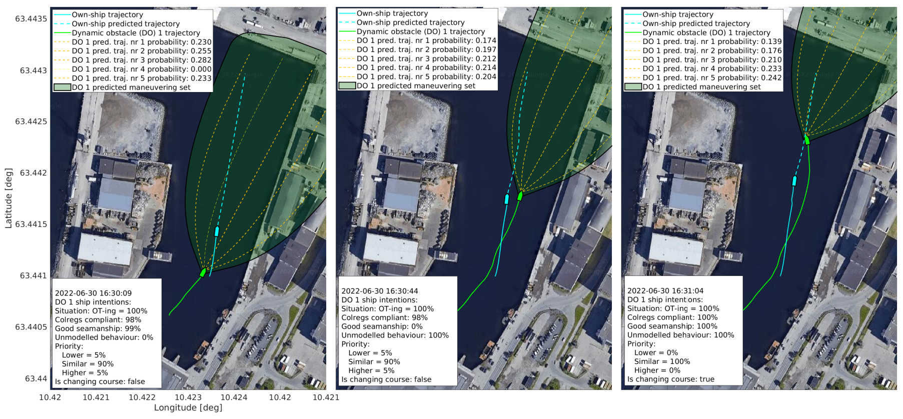
(d) Own-ship heading and speed.

Figure 7.12: Scenario 6 - The own-ship overtakes a dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

diction setup, where obstacles are predicted to follow alternative trajectories about a nominal straight line from their current course and speed when the COLREGs situation starts. The waypoints set for the nominal straight line path of DO 1 did here not properly reflect the ground truth planned trajectory of the obstacle, as it changed course northwards when starting the overtaking maneuver. An improved approach would be to update the dynamic obstacle nominal straight line trajectory at regular intervals, especially after a COLREGs situation has ended. This would make the alternative prediction scenarios better reflect the possible maneuvering areas in the vicinity of the obstacle.

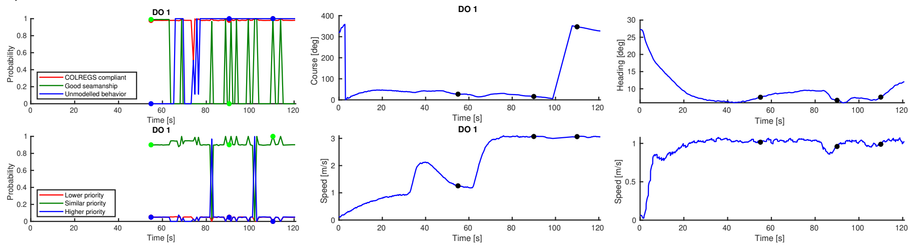
Scenario 8 - Combined Overtaking and Crossing Port Side

Results from the first three-ship scenario is given in Figure 7.14. Milliampere 2 has a stand-on role towards Havfruen (DO 1), and a give-way role towards the Cyberotter (DO 2). The PSB-MPC shows compliance with COLREGs rule 17(d) by ignoring its stand-on role and performing an evasive starboard maneuver. The intention inference module is able to estimate that Havfruen will make a give-way maneuver by passing behind both ships. For the Cyberotter, the intention model starts by correctly predicting that it will keep its course and speed. At the



Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
DO 1 pred. traj. nr 1 probability: 0.230
DO 1 pred. traj. nr 2 probability: 0.255
DO 1 pred. traj. nr 3 probability: 0.282
DO 1 pred. traj. nr 4 probability: 0.000
DO 1 pred. traj. nr 5 probability: 0.233
DO 1 predicted maneuvering set
2022-06-30 16:30:09
DO 1 ship intentions:
Situation: OT-ing = 100%
Colregs compliant: 98%
Good seamanship: 99%
Unmodelled behaviour: 0%
Priority:
Lower = 5%
Similar = 90%
Higher = 5%
Is changing course: false
DO 1 pred. traj. nr 1 probability: 0.174
DO 1 pred. traj. nr 2 probability: 0.197
DO 1 pred. traj. nr 3 probability: 0.212
DO 1 pred. traj. nr 4 probability: 0.214
DO 1 pred. traj. nr 5 probability: 0.204
2022-06-30 16:30:44
DO 1 ship intentions:
Situation: OT-ing = 100%
Colregs compliant: 98%
Good seamanship: 0%
Unmodelled behaviour: 100%
Priority:
Lower = 5%
Similar = 90%
Higher = 5%
Is changing course: false
DO 1 pred. traj. nr 1 probability: 0.139
DO 1 pred. traj. nr 2 probability: 0.176
DO 1 pred. traj. nr 3 probability: 0.210
DO 1 pred. traj. nr 4 probability: 0.233
DO 1 pred. traj. nr 5 probability: 0.242
2022-06-30 16:31:04
DO 1 ship intentions:
Situation: OT-ing = 100%
Colregs compliant: 100%
Good seamanship: 100%
Unmodelled behaviour: 100%
Priority:
Lower = 0%
Similar = 100%
Higher = 0%
Is changing course: true
Latitude [deg]
Longitude [deg]

(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes.



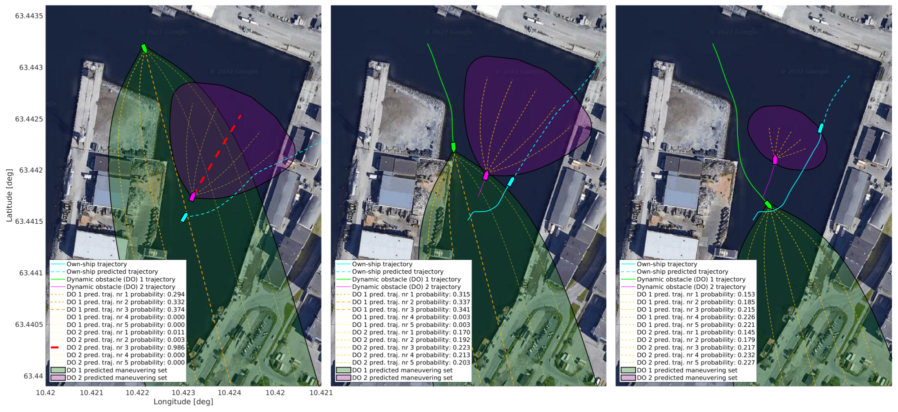
(b) Dynamic obstacle intention states. **(c)** Dynamic obstacle course and speed. **(d)** Own-ship heading and speed.

Figure 7.13: Scenario 7 - The own-ship is overtaken by a semi-compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

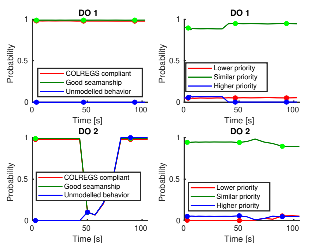
time $t = 50\,\text{s}$ the intention module notices that the Cyberotter changes its course towards starboard. As this is a change in course towards a collision, the intention module concludes with unmodelled behavior.

Scenario 9 - Combined Crossing Starboard Side and Port Side

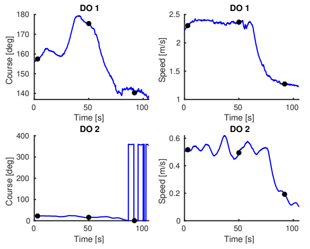
Results from the second three-ship scenario are given in Figure 7.15. The own-ship running the COLAV system initially slows down and then changes its course to pass behind the Cyberotter. The intention model initially predicts correctly that the Cyberotter will keep its course and speed, while Havfruen will cross behind the own-ship. A bit before the 50 s mark the Cyberotter is observed changing its course towards a collision, which makes the intention model mark it as unmodelled behavior.



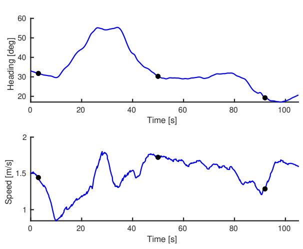
(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes. The Havfruen vessel (DO 1) is shown in green, whereas the Cyberotter (DO 2) is shown in purple.



(b) Dynamic obstacle intention states.



(c) Dynamic obstacle course and speed.



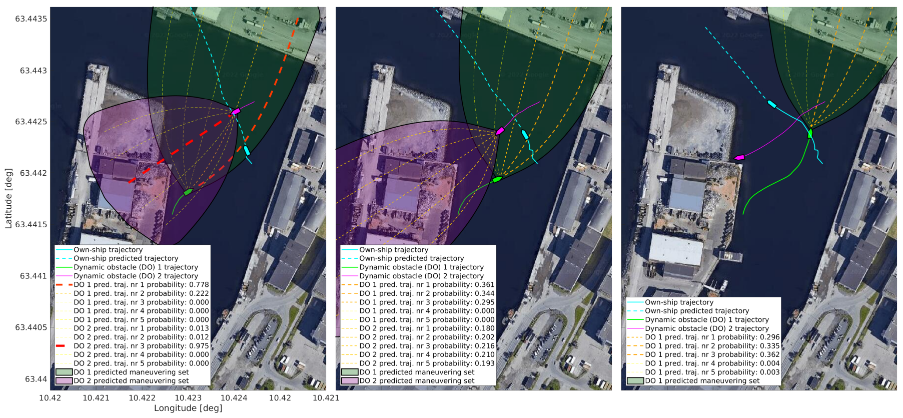
(d) Own-ship heading and speed.

Figure 7.14: Scenario 8 - Combined overtaking and crossing port side scenario with compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

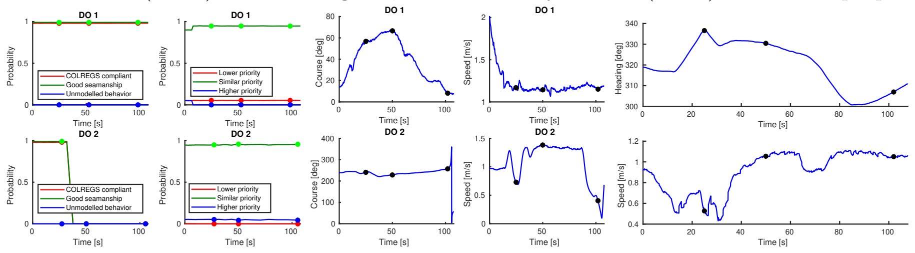
7.6 Discussion

7.6.1 Experiment Outcome

From gauging the experimental results, the COLAV system using the PSB-MPC as a deliberate planning algorithm with a DBN for intent information shows promise and results in increased situation awareness for the own-ship. In Scenario 4 the intention-aware COLAV system enables the own-ship to keep its course and speed, as it predicts that the obstacle will perform an evasive maneuver. When it is apparent, in Scenario 5, that the obstacle will not perform an evasive maneuver, the COLAV system makes use of its intention model to better avoid collision. In Scenario 2 the intention module estimated that the dynamic obstacle was not adhering to the COLREGs, which enabled the PSB-MPC to ignore COLREGs as well to plan a collision-free trajectory. In the trials, the own-ship running the COLAV system is shown to comply with the COLREGs rules 8 and 13-17. Furthermore, better adherence with COLREGs rule 7 is also achieved, as the intention module enables better collision risk assessment and COLREGs situation evaluation in the


Latitude [deg]
Longitude [deg]
Own-ship trajectory
Own-ship predicted trajectory
Dynamic obstacle (DO) 1 trajectory
Dynamic obstacle (DO) 2 trajectory
DO 1 pred. traj. nr 1 probability: 0.778
DO 1 pred. traj. nr 2 probability: 0.222
DO 2 pred. traj. nr 3 probability: 0.975
DO 1 predicted maneuvering set
DO 2 predicted maneuvering set

(a) Situation plot at multiple time instants. The vessels are scaled for visualization purposes. The Havfruen vessel (DO 1) is shown in green, whereas the Cyberotter (DO 2) is shown in purple.



(b) Dynamic obstacle intention states. **(c)** Dynamic obstacle course and speed. **(d)** Own-ship heading and speed.

Figure 7.15: Scenario 9 - Combined Crossing Starboard Side and Port Side scenario with compliant dynamic obstacle (DO). The dots on the lower subfigures show the values at the three time instants in (a).

PSB-MPC planning algorithm.

7.6.2 Uncertainty Management

As the own-ship uses GNSS with a VRS for real-time corrections, the navigation data for the ferry has negligible uncertainty compared to the safety margin (d_{safe}) set for the ferry. On the other hand, no corrections were applied to the GNSS data from the two dynamic obstacles, and we thus relied on the KF performing adequately. The filter was tuned such that positional estimates with standard deviations around 0.7 m were obtained, which was verified to be correct before the experiments started. Furthermore, the kinematic uncertainty associated with dynamic obstacle estimates was handled through the collision probability estimation in the PSB-MPC, as in [20].

Regarding the map data used for avoiding grounding hazards in the PSB-MPC, a manual drive-through of the Nyhavna basin boundary was done to verify the data accuracy. As the map data did not include newer static obstacles such as the

Havet sauna platform, a new landfilling on the western side, and a few docked vessels along the basin boundary, unmapped hazards were added manually before the experiment.

7.6.3 Limitations

There are a few limitations to the experiments in this work. Firstly the collision avoidance algorithm was designed for ships meeting on open seas, where COLREGs is normally considered. Due to the limitations of the Miliampere 2 ferry, the experiments had to be done in inland areas where the sea was sufficiently calm. The largest available area in Trondheim was quite small making it difficult to realistically test the algorithm. Secondly, the Milliampere 2 platform with its commercial DP-system was not designed for agile ship maneuvering, and this put a limitation on the performance one could extract from the vessel. The DP-system was overdamped, tuned for passenger comfort and small, slow movements as a ferry should perform. The vessel slows down substantially when performing heading maneuvers, which made the trajectory tracking challenging. For the overtaking scenarios, the own-ship had problems keeping to its speed due to the platform being designed and tuned for keeping speeds of nominally 1.0 m/s.

These concerns made the limitations of the intention inference module affect the experiments more than we would expect it to do in open waters. With low velocities, the dynamic obstacles had problems keeping a constant course. As the intention inference module interprets all course changes larger than a threshold as deliberate actions taken by the dynamic obstacle, it often had to default to the unmodelled behavior state to explain its observations. This was especially apparent when using the Cyberotter (Scenario 8 and 9) and for Havfruen in the overtaking scenario (Scenario 6) when its speed was reduced to 1 m/s. One could increase the course change threshold to filter away more of the random course changes. However, this has the drawback of making the module potentially miss actual course changes that it should consider. A smarter method for ignoring random motions is therefore warranted. Furthermore, the intention model does not consider land and static obstacles. This was especially noticeable in the overtaken scenario (Scenario 7) when the dynamic obstacle changes its course to avoid collision with a floating platform.

Other limitations with the intention model are with the approach for handling measurements in the middle of the course change and the approach for limiting the computational burden. The approach presented in Equation (7.15) for handling measurements in the middle of the course change does not realize that a ship is incompliant during a course change. This makes the model react unnecessarily slow when the ship is obviously changing course in the wrong direction. This could be solved by considering whether it's possible for the ship to cross in a compliant manner by continuing to change course in the same direction. The limitations used on deciding when new time steps are introduced were set quite conservatively to ensure that the computational burden would under no circumstances be too large for real-time inference during the experiments. Less conservative bounds could probably have been chosen. A better approach is warranted that can guarantee acceptable computation time while at the same time keeping more data in memory. This could for example be done by implementing the sliding window approach as discussed in Chapter 6.

While doing the experiments it became apparent that being able to use the heading of the other ships would be better than using the course. While supervising the experiments on board Milliampere 2 we could see that the heading of the dynamic obstacle had changed long before it became apparent when looking at the course alone. Enabling an autonomous ship to track the heading of the other ship, by for example using extended object tracking [168], could enable the inference module to more quickly realize what the other ship is doing and thereby enable the collision avoidance module to respond quicker.

7.7 Conclusion

In this work, a dynamic obstacle intention-aware PSB-MPC-based COLAV system has been presented, using a DBN for intention inference online. The resulting system is verified in experimental trials to show compliance with COLREGs rules 8, 13-17. The results presented show that incorporating a way of inferring the intentions of nearby dynamic obstacles proves to give COLAV planners such as the PSB-MPC improved situation awareness. This results in more efficient trajectory planning, where COLREGs-compliant maneuvers can be aborted if the other vessel is shown or estimated to be non-compliant. It also better enables stand-on compliance for autonomous agents, as it is necessary to infer to which degree a give-way vessel will perform proper maneuvering.

Future work involves testing the intention-aware COLAV system in a more open sea environment, as the intention inference algorithm was originally not designed for situations in confined spaces with limited maneuvering possibilities. The presented COLAV system should also be tested in a less pre-planned setting, with target tracking in real maritime traffic. Further developments are needed to make the intention inference more robust to natural variations in the measured course of other ships and to make it handle course changes and limiting the computational burden in a better manner. Furthermore, work is needed to improve the prediction of alternative obstacle maneuvering scenarios to also consider nearby grounding hazards and the current COLREGs situation.

Acknowledgments

The authors want to thank Egil Eide and Bjørn-Olav Holtung Eriksen for support during the experiments, Helene Engebakken Haugen, Guro Drange Veglo and Peder Hofstad Lycke for operating the Havfruen vessel, and Øystein Volden and Petter Solnør for operating the Cyberotter.

Chapter 8

Validation using historical AIS data

This chapter is based on the following publication

[54] **S. V. Rothmund**, H. E. Haugen, G. D. Veglo, E. F. Brekke, and T. A. Johansen, “Validation of ship intention model for maritime collision avoidance control using historical AIS data,” *Submitted to ECC*, 2023

Software for running the intention model on AIS data is developed by H. E. Haugen with modifications from S. V. Rothmund. Distributions based on AIS data are developed by G. D. Veglo. Preparation of results was performed by S. V. Rothmund. Supervision was provided by S. V. Rothmund, E. F. Brekke, and T. A. Johansen. The first draft of this chapter was written by S. V. Rothmund and revised by T. A. Johansen and E. F. Brekke.

8.1 introduction

In Chapter 6 the intention model is tested on different simulated scenarios where the goal is to illustrate how the model works. In Chapter 7 the intention model is used in controlled experiments where it is demonstrated how the model worked on real measurements and in a real-time setting. In this chapter, the intention model is tested on historical AIS data gathered near the coast of Norway. Furthermore, some of the example distributions used by the intention model in Chapters 6 and 7 are in this chapter replaced with empirical distributions extracted from the historical AIS cases. The contribution of this chapter is to test how the intention model performs on real ship encounters with real distributions, thereby validating the model and identifying potentials for improvement.

The rest of this chapter is structured as follows. First, an overview of the AIS data set is given in Section 8.2. Then the intention model used by this chapter is specified in Section 8.3. Section 8.4 presents the distributions used by the mode.

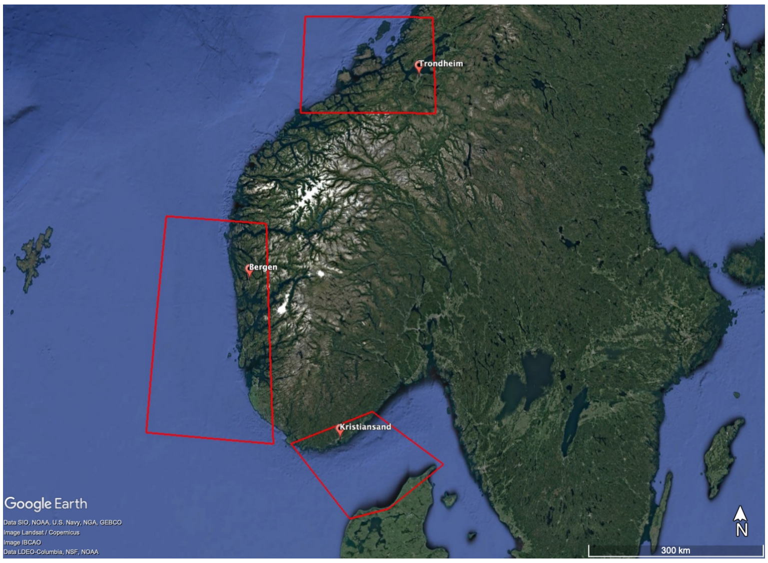

Trondheim
Bergen
Kristiansand
Google Earth
Data SIO, NOAA, U.S. Navy, NGA, GEBCO
Image Landsat / Copernicus
Image IBCAO
Data LDEO-Columbia, NSF, NOAA
N
300 km

Figure 8.1: Areas on the coast of Norway where AIS data has been collected [169].

The results of applying the model to historical AIS cases are given in Section 8.5 and discussed in Section 8.6, before a conclusion is given in Section 8.7.

8.2 AIS data set

This chapter uses a AIS data set provided by The Norwegian Coastal Administration (Kystverket) collected from there different parts outside of the coast of Norway shown in Figure 8.1. The northern data set has data points from between 1.1.2018 and 31.7.21, except between 1.6.2019 and 31.7.2019. The southern data set from 1.1.2018 to 31.12.2019 and the western data set from 1.1.2019 to 31.12.2020.

The data set consists of the AIS messages sent from all ships with AIS transponders in the area within the specified time periods. The most important part of the AIS message for this chapter is the time-stamp, the unique identifier of each ship, their position, course, and speed over ground. Note that, generally, only larger ships are required to have AIS transponders [141]. This means that there might be ships present but not in the data set, which affected the behavior of the logged ships.

This data set has been previously analyzed by multiple master [170]–[172] and doctoral thesis [169] at NTNU. From these works, a refined data set is produced where 28421 encounter cases are extracted and automatically classified with different parameters of interest. Of these, 1206 encounter cases are manually classified to actually be collision avoidance situations where the ships act in accordance with COLREGs.

As the different ships do not transmit AIS messages with the same frequency or

Table 8.1: The initial distribution used for the intention variables. $\mathcal{N}(\mu, \sigma)_{[a,b]}$ indicates a normal distribution with expected value μ, standard deviation σ, truncated to be between a and b, and discretized into 30 evenly spaced intervals. The probability of "true" is given for binary states.

Symbol	Description	Prior
$\mathcal{I}_{AT}$	Ample time	See Figure 8.3(a)
$\mathcal{I}_{CC}$	COLREGS compliant evasive maneuvers	0.99
$\mathcal{I}_{GS}$	Good seamanship	0.99
$\mathcal{I}_{P_\rho}$	Priority	$[\mathit{higher} = 0.05,\ \mathit{similar} = 0.90,\ \mathit{lower} = 0.05]$
$\mathcal{I}_{RC}$	Risk of collision distance	$\mathcal{N}(1500\,\mathrm{m}, 250\,\mathrm{m})_{[0,2500]}$
$\mathcal{I}_{RCF}$	Risk of collision distance front	$\mathcal{N}(1500\,\mathrm{m}, 250\,\mathrm{m})_{[0,2500]}$
$\mathcal{I}_{SD}$	Safe distance	See Figure 8.3(b)
$\mathcal{I}_{SDF}$	Safe distance front	See Figure 8.3(b)
$\mathcal{I}_{SDM}$	Safe distance midpoint	See Figure 8.3(b)
$\mathcal{I}_{SS}$	Situation start distance	$\mathcal{N}(13\,000\,\mathrm{m}, 1000\,\mathrm{m})_{[0,15000]}$
$\mathcal{I}_{U}$	Unmodeled behaviour	0.00001

phase, interpolation was done by [170] to get the estimated position, course, and speed of both ships at the same time points. The resulting data set has a time resolution of one message per minute. More information on the data set can be found in [170], [172].

The data available for the research presented in this chapter is therefore:

- One file for each encounter case, consisting of the timestamps, id, position, speed, and course of all ships in the encounters at different time steps. Only the interpolated cases where the data are synchronized are considered in this work.
- A list over all encounter cases and automatically extracted parameters.

8.3 Model

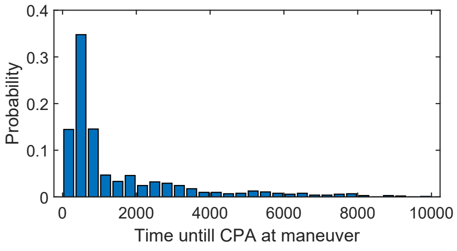
This chapter uses the full model containing situation started and risk of collision, as in chapter Chapter 6, with the improvements proposed in Chapter 7. More specifically the model in Chapter 6 is used with the improvements proposed in Equations (7.13) to (7.17).

8.4 Definitions and prior distributions

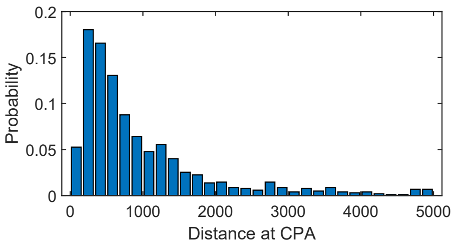
Initial distributions are given in Table 8.1 while parameter choices are given in table 8.2.

Table 8.2: Parameter choices.

Symbol	Description	Value
$\mathcal{P}_{CIC}$	Max change in course that is considered as keeping the course	15 degree
$\mathcal{P}_{CIS}$	Max change in speed that is considered as keeping the speed	1.5 m/s

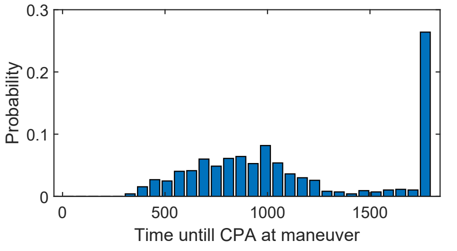


(a) Distribution for "ample time".

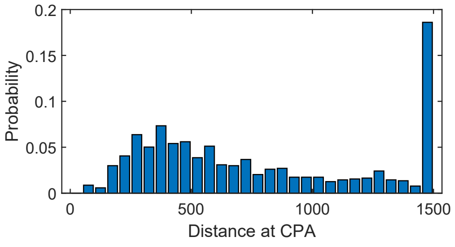


(b) Distribution for "safe distance".

Figure 8.2: Empirical distributions extracted from historical AIS cases from outside the Norwegian coast. This figure shows how the distribution behaves for large values.



(a) Distribution for "ample time".



(b) Distribution for "safe distance".

Figure 8.3: This figure shows the same data as in Figure 8.2 but where the focus is placed on shorter distances and times that are more important to distinguish for collision avoidance. These distributions are used by the intention model.

The distributions for "ample time" and "safe distance" are evaluated based on parameters extracted from the AIS data set by [169]–[172]. Only the cases that are manually inspected and classified as being correct COLREGs situations are used. The distributions showing the full width of the data are shown in Figure 8.2. Figure 8.3 shows the same data but where maximal ample time is set to 1800 s and maximal safe distance to 1500 m. This ensures a large resolution on the times and distances relevant for collision avoidance. All values larger than the maximum are added to the largest bin.

The distribution for "risk of collision distance" and "situation start distance" are not as easy to extract from the classified data. These are instead manually chosen to work well for the cases used to generate the results.

Including the course and speed of ships as nodes in the DBN requires discretizing the state. Discretization steps of $10°$ are chosen for the course and $0.72\,\mathrm{m/s}$ for speed (with an upper limit on $18\,\mathrm{m/s}$). This was the finest resolution that was feasible to use when discretizing using the GeNie software [125]. The course and speed change parameters given in Table 8.1 had to be chosen larger than their corresponding discretization step size.

The discrete states represent alternative explanations to acting in accordance with COLREGs. The intention model weighs the probability of these states being in an adverse state against the probability of the ship having a definition of ample time, safe distance, and risk of collision lower than the current measured value. Using very small values may cause numerical inaccuracies as a sampling-based approximate solver is used for evaluating the DBN. For unmodelled behaviour, the lowest possible value is chosen that did not cause any problems.

8.5 Result

This section presents the results of applying the intention model to different encounter cases from the AIS data set. Each encounter is played back with the intention inference module active. The estimated intentions for both ships in the encounter are presented. The encounter cases are manually chosen from the data set. Focus was placed on testing encounters where the main maneuvers of the ship relate to collision avoidance, and cases where it is interesting to infer the intentions.

For each case it is shown how the following parameters develop over time:

- “Speed over ground”: Directly taken from the AIS data.
- “Course”: Directly taken from the AIS data.
- “Intention COLREGs situation”: Evaluated based on what the value of the classifiers given in Figures 6.2 and 6.3 are at the moment when the situation starts.
- “Predicted distance of at CPA”: Evaluated assuming both ships keeping their current course and speed.
- “Change in course port/starboard”: Indicates whether the reference ship has significantly changed course relative to the initial course.
- “Situation started”: This state defines when, among others, the COLREGs situation and initial course is specified. Distance between the ships is used to define when the situation starts.
- “Risky situation”: Indicates whether there has been a risk of collision, in which case the behavioural rules from COLREGs should be followed for the rest of the encounter. Distance at CPA is used to define whether there is a risk of collision.
- “Intention COLREGs compliant evasive maneuver”: Indicates whether the ship appears to consider COLREGs when performing evasive maneuvers.

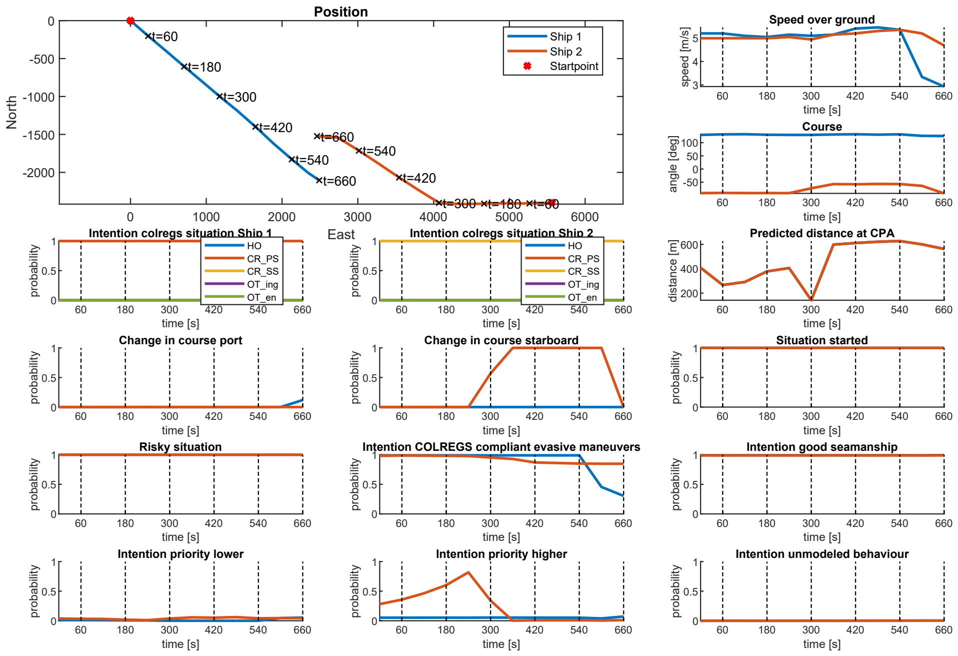


Figure 8.4: Case 1 - Correct crossing situations. The dashed vertical lines correspond to the timestamps in the position plot.

This state does not change if the ship does not perform an evasive maneuver, which will instead affect intention priority higher.

- "Intention good seamanship": Indicates that the ship is not changing which side it is performing an avoidance maneuver towards and is not changing its course to cross at a shorter distance.
- "Intention priority lower": Indicated that the ship will give-way no matter the COLREGs situation.
- "Intention priority higher": Indicates that the ship will stand-on no matter the COLREGs situation.
- "Intention unmodelled behaviour": Indicates that the ship acts in a manner not considered by the intention model.

Case 1 - Crossing correctly

Figure 8.4 shows two ships meeting in a crossing encounter. The red ship performs a COLREGs-compliant evasive maneuver at the 300 s mark. The higher priority intention of the red ship increases quite a lot before the maneuver as the maneuver is quite late compared to the ample time distribution. The probability that the blue ship will comply with COLREGs when performing evasive actions falls at the last two time steps due to the blue ship reducing its speed by 40% and turning its course a bit to port.

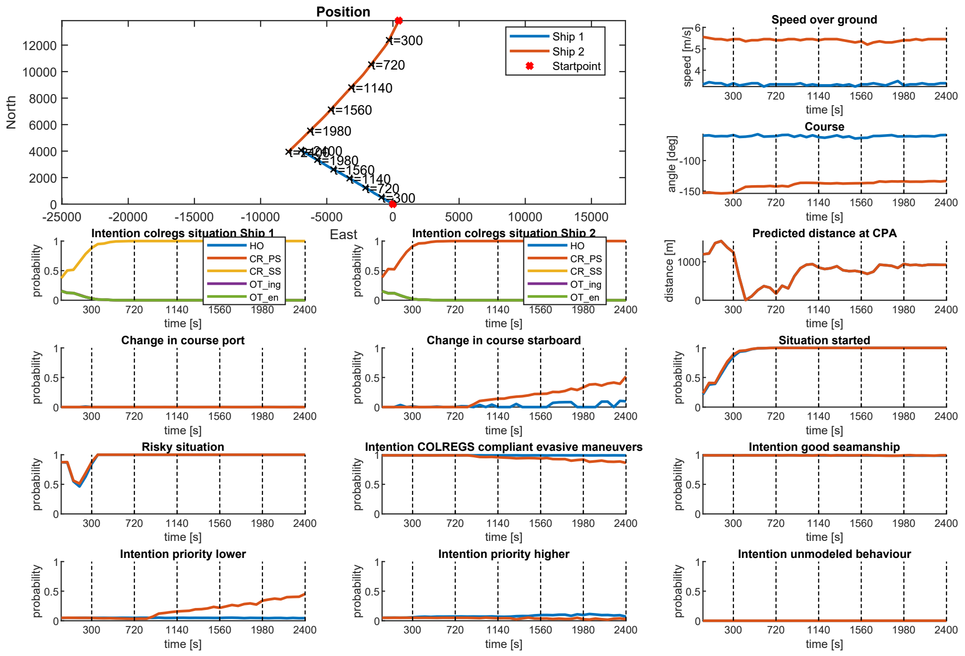


Figure 8.5: Case 2 - Crossing situation where the wrong ship acts. The dashed vertical lines correspond to the timestamps in the position plot.

Case 2 - Crossing incorrectly

Figure 8.5 shows a crossing encounter where the ship that according to COLREGs rule 15 has the stand-on role performs an early evasive maneuver. The course change is quite slow and small making the intention model evaluate a medium probability that the red ship has performed a starboard evasive maneuver and therefore acts as if it has a lower priority. The probability that the red ship intends to follow COLREGs when performing evasive maneuvers does not reduce much as the maneuver is in compliance with COLREGs rule 17c. The probability that the blue ship acts as if it has higher priority does not increase significantly even though it does not change its course or speed. This is due to the red ship already having acted which makes keeping the course and speed acceptable give-way behavior for the blue ship.

Case 3 - Crossing or head-on

Figure 8.6 shows a situation where the ships are approaching at an angle close to the border between crossing and head-on. The intention model evaluates a 70% chance that it is a crossing situation and a 30% that it is a head-on situation. Only the red ship acts by taking a COLREGs compliant starboard maneuver. The resulting behavior is compliant with the rules for both crossing and head-on situations making the COLREGs situation intentions not change.

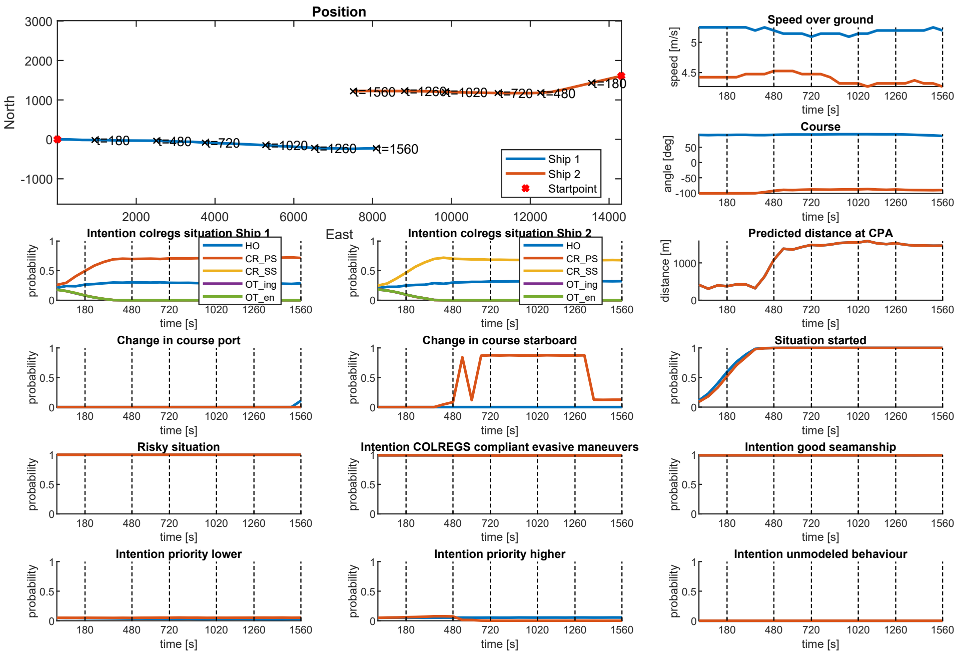


Figure 8.6: Case 3 - Situation which can be interpreted as both a crossing and a head-on situation. The dashed vertical lines correspond to the timestamps in the position plot.

Case 4 - Head-on passing on starboard side

Figure 8.7 shows a head-on situation where the ships pass with each other on the starboard side which is contrary to COLREGs rule 14. The intention model infers that the red ship is as acting as if it has higher priority as it does not change its course or speed. The intentions of the blue ship are not as clear as it changes its course a bit back and forth, making the intention model switch between the ship acting as if it has a higher priority, performing an incompliant evasive maneuver, and showing unmodeled behavior. In all cases, good seamanship is quite reduced.

Case 5 - Head-on low risk of collision

Figure 8.8 shows a similar situation as in Case 4 but where the ships pass at a greater distance. In this case, the behavior is mainly explained by there not being a risk of collision, in which case the ships do not need to act. At the start of the encounter, the model evaluates around a 75% chance that the ships consider that there is a risk of collision. As the ships get close without performing an evasive maneuver this probability gradually decreases.

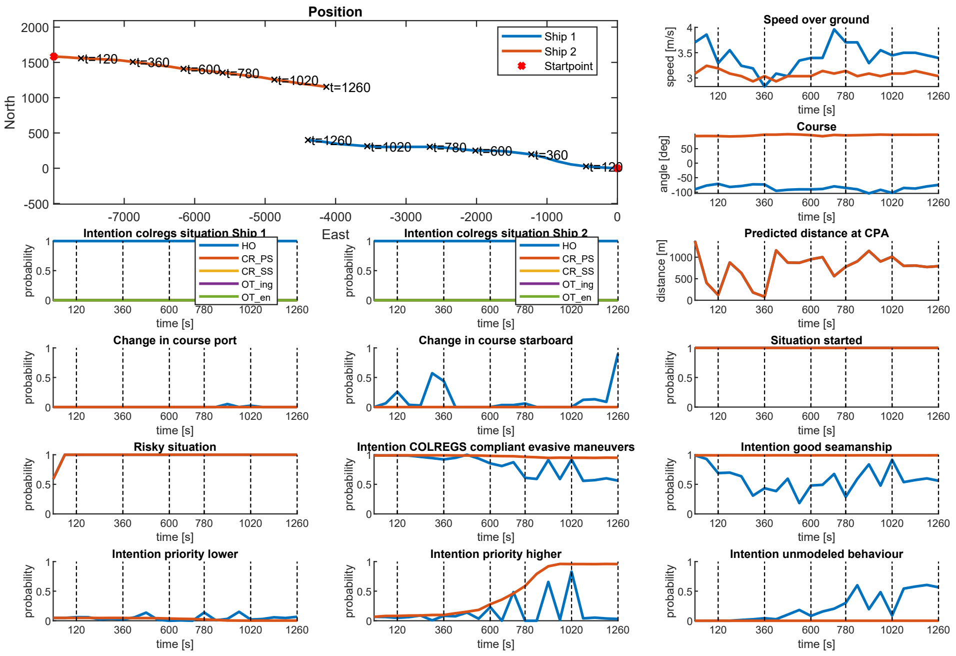


Figure 8.7: Case 4 - Head-on situation where the ships pass with each other on the starboard side. The dashed vertical lines correspond to the timestamps in the position plot.

Case 6 - Head-on port maneuver

Figure 8.9 shows a head-on situation where the ships actively change course to pass with the other on the starboard side, which is contrary to COLREGs rule 14a. The model correctly identifies that the ships are performing evasive maneuvers in a COLREGs incompliant manner.

Case 7 - Situation started

Figure 8.10 shows a situation where uncertainty about when the situation starts plays an important role. Before the 360s mark, the blue ship takes a starboard turn before it aligns back up on a collision course. The intention model assumes that the COLREGs situation started after this course change, after which the blue ship performs a late COLREGs incompliant evasive maneuver by changing its course to cross in front of the red ship. To get this result the situation start distance had to be reduced to $\mathcal{N}(5000, 750)_{[0,8000]}$. Without this change then the blue ship was marked as acting in an unmodelled manner as it turned towards a collision course after the situation had started.

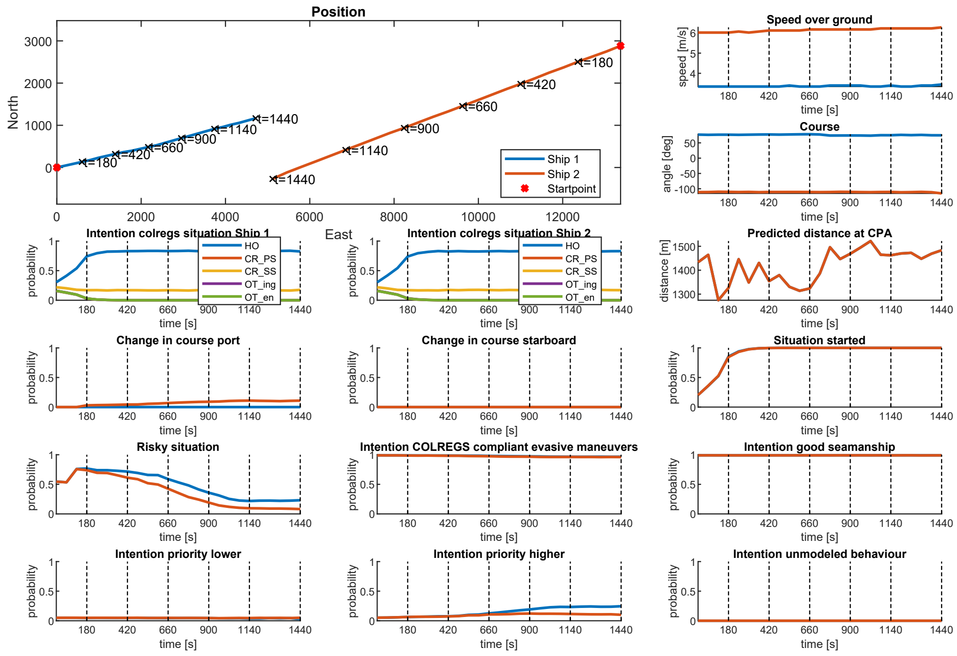


Figure 8.8: Case 5 - Head-on situations where the ships pass on the wrong side at a large distance. The dashed vertical lines correspond to the timestamps in the position plot.

8.6 Discussion

The results show how the intention model manages to correctly identify when ships act in accordance with COLREGs, Cases 1 and 3, and when this is not the case, Cases 6 and 7. Furthermore, they show that the intention model can distinguish between incompliant maneuvers, Cases 6 and 7, compliant maneuvers but the wrong ship acts, Case 2, and situations where no ship acts, Cases 4 and 5.

The “ample time” distribution defines when ships, that have so far not performed a maneuver, will be marked as either having a higher priority, Case 4, or considering that there is no risk of collision, Case 5. In Cases 1 and 7 the probability that one of the ships acts as if it has a higher priority increases quite a lot before the evasive maneuver is performed. It might be that this apparently late behavior should be expected if the type of ships and location of the encounter is also considered. Having separate distributions for different locations and ship types can enable the intention model to understand that, for example, small ships in inland areas tend to act later and come closer than large ships in open waters.

A limitation with the “ample time” definition is that it does not consider how difficult it is to perform the evasive action. Much earlier actions must be taken in head-on situations where there is a large offset to the wrong side, Case 5, than in for example crossing situation with no offset, Case 1.

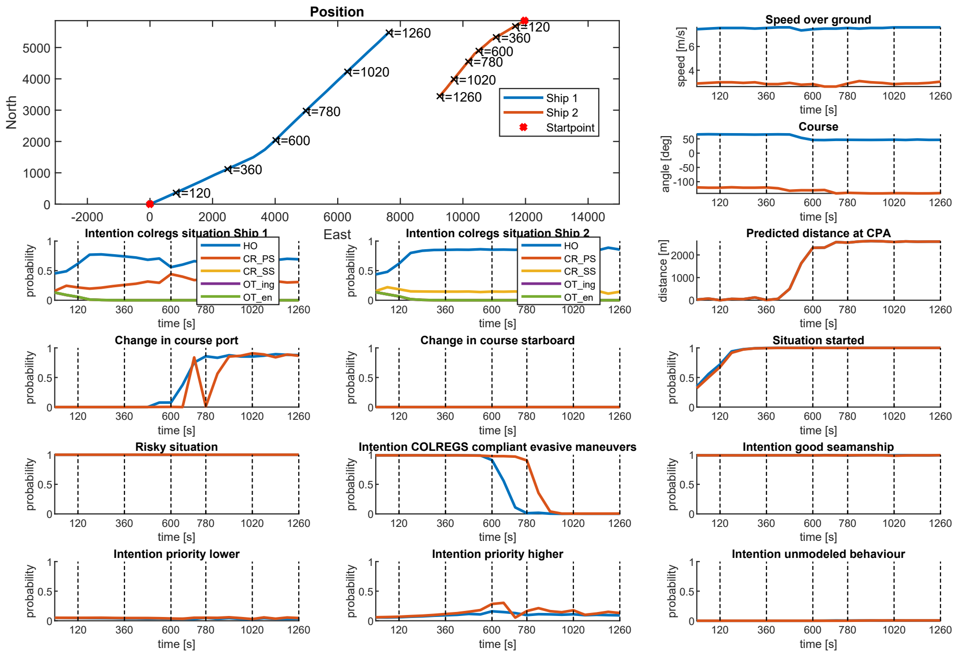


Figure 8.9: Case 6 - Head-on situation where the ships act to pass on the incorrect side according to COLREGs. The dashed vertical lines correspond to the timestamps in the position plot.

Cases 4 and 5 shows the effect of the “risk of collision” distribution. Both present cases where neither ship performs an avoidance action. In Case 4 this is explained by the ship acting as if it has higher priority, while it in Case 5, where the ships pass at a larger distance, is explained by there not being a risk of collision.

The effect of the “situation start” distribution is shown in Case 7. Reducing the situation start distribution enabled the intention model to filter out a course change that was not related to collision avoidance. Having this low definition of when the situation starts would not work in other cases, such as Case 2, as it would then not realize that the red ship has actually performed an avoidance action. It might be that a different definition of when the situation starts is warranted in these two cases if one is for example near the coast while the other is in open waters. Alternatively, it might be inadequate to define when the situation starts from distance alone.

Modeling the “situation start” in the DBN has some advantages as shown in Case 7, but makes the model much more complex and requires that the initial heading and speed of the ship is included as nodes in the DBN. Introducing them as nodes requires that the heading and speed are discretized. The maximum number of discretization intervals that can be used in practice are quite limited as the computational burden of evaluating a DBN increases substantially with the num-

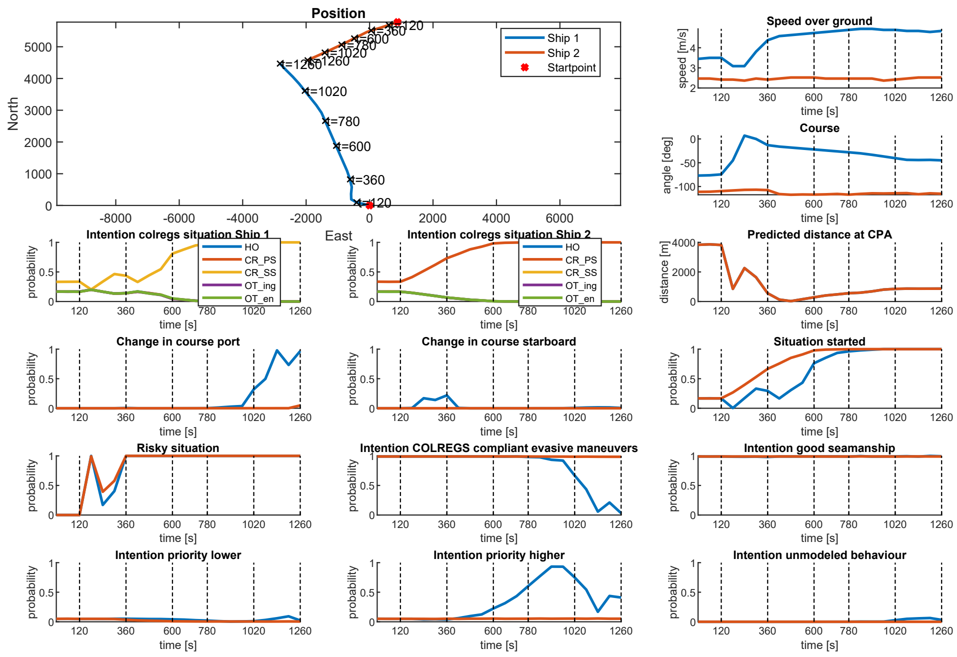


Figure 8.10: Case 7 - Crossing situation where there is a maneuver change unrelated to collision avoidance at the start of the encounter. The dashed vertical lines correspond to the timestamps in the position plot.

ber of intervals. Having few and large intervals is a problem when a ship starts with a course or speed close to the boundary between two intervals, then a small change in course in one direction can be marked as the ship changing course, while a larger course change in the opposite direction can be marked as the ship keeping its course. To limit this problem a quite large definition of how small speed and course change that are considered as standing-on is used. This is especially helpful in Case 5 where even though the red ship keeps a very steady course, the probability that the ship changes its course to port starts to increase. This large definition has limitations in Case 2 where it causes uncertainty about whether or not the red ship has performed a collision avoidance maneuver, even though it clearly has.

Cases 1 and 2 show an error in the inference where the probability that the ships will comply with COLREGs when performing evasive actions decreased a bit, even when the avoidance actions are in accordance with COLREGs. This error does not constitute a large problem as the reduction in probability is small, but it is an indication that something is wrong with the inference. The cause of this error has not been found, but it seems to not directly be related to the logic of the intention model itself. It rather seems to be related to the model being sensitive to numerical errors. Further investigation on understanding and alleviating this error is needed.

Another limitation observed during testing, but not present in any of the presented cases, is that having a safe distance to the current midpoint in head-on situations, as defined in Equation (6.12), is not enough if the ships meet at a relatively large angle. This can be solved by requiring that the ships in addition must have a safe distance (SD) and pass at the correct side ($\mathcal{M}_{MPS_\rho}[t] == port$) at CPA.

8.7 Conclusion

In this chapter, the intention model presented in Chapter 6 is tested on historical data using empirical distributions. The tests demonstrate that the model is able to identify different ways the ship's behavior can conflict with the rules, either by acting as if they have higher or lower priority, giving way in an COLREGs incompliant manner, or by not considering it a risk of collision. It furthermore demonstrates that it is feasible to use empirical distributions in the model.

This research has also demonstrated some weaknesses with the model that opens up for further work. The main limitation is with filtering out actions unrelated to collision avoidance. The current implementation that considers whether the situation has started can handle some cases but makes the model more complicated and introduces discretization issues. Finding a better solution would substantially improve the model. Furthermore, a new method for ample time that considers how difficult the avoidance maneuver will be, is warranted. Lastly, the model does not handle actions not related to collision avoidance, such as avoiding grounding or following a traffic separation scheme or narrow channel.

Acknowlegement

We would like to thank the Norwegian Coastal Administration (Kystverket) for access to the AIS data set.

Part III

Concluding remarks

Chapter 9

Conclusion

This chapter consists of three sections. First, in Section 9.1, a summary and reflection is given where the focus is placed on differences and similarities between the chapters. Then the discussion on the contributions of the different chapters to the research questions and opportunities for future work are presented in Section 9.2. Finally, some concluding remarks are given in Section 9.3.

9.1 Summary and reflections

This thesis presents different contributions on increasing the risk awareness for control of autonomous robots. The thesis consists of two parts each with its own focus. Part I focuses on a drone operating in static environments where the challenge is to identify the state of the system and the environment based on incomplete information. Part II focuses on an autonomous ship in dynamic environments with multiple agents. Here the challenge is to identify the intentions of the other agents.

Chapter 3 and Part II present contributions on increasing risk awareness for robotic systems by improving the systems model of single factors that are connected with significant uncertainty. Chapter 3 considers the uncertainty in the location and shape of obstacles stemming from observing the environment with a wide-angle radar. This chapter develops a collision avoidance algorithm that considers this uncertainty together with the uncertainty in the drones motion to evaluate the risk of collision. Considering the uncertainty enables the system to be proactive when entering areas where its knowledge is incomplete. Part II considers the uncertainty in the intention of other ships at sea. This uncertainty is arguably the largest source of uncertainty in encounter situations. This part develops an intention model that enables the system to identify situations prone to cause misunderstanding, understand the intention of other ships based on their observed behavior, and adapt the model to the current situation.

Chapters 4 and 5 presents two different contributions on how to give a robotic system a holistic understanding of risk. Chapter 4 considers operational decision-making that has to be done every time a task execution fails. This chapter develops a decision-making system that is used to decide whether a task should be attempted again, the task should be skipped, or maintenance should be requested. By having a holistic model that considers the history of the outcome of actions, the system is able to identify the underlying causes which enable it to take appropriate action. Considering the uncertainty in the situation awareness enables the system to proactively avoid hazardous events. Chapter 5 considers decisions that have to be made continuously during an operation. This chapter develops a holistic risk model that monitors the state of the system and changes safety-critical parameters to ensure a safe operation. The holistic model enables the supervisory risk controller to identify the underlying cause by combining information over time and to see different problems in light of each other when making decisions.

A difference in the proposed method of Chapters 4 and 5 is how the underlying states are defined. In Chapter 4 focus is placed on which states that can be distinguished from each other based on the available information, that is, states which are observable. In contrast, Chapter 5 instead focuses on the results of the risk analysis when choosing a state. In practice, this difference is not big as it is still useful to have the result of a risk analysis available in Chapter 4 when considering which causal factors that are relevant to include, and since the detail level of causal factors in Chapter 5 should be based on which states that are observable.

A more substantial difference is in how the underlying states are combined to evaluate the risk. In Chapter 4 it is assumed that the underlying causal factors could be realized as either sufficient or insufficient on each task execution attempt. Logical gates, such as *and* and *or* are then used to combine the causal factors to evaluate the risk. This approach moved all uncertainty to the realization of the underlying causes. A drawback of this approach is that it does not handle cases where multiple underlying factors affect the realization. This is handled in Chapter 5 by allowing more freedom in the modeling of the resulting risk. A drawback with this approach is that it requires substantially more quantification work than in Chapter 4.

The difference in the formulation also relates to Chapter 4 considering binary outcomes of the hazardous event occurring or not, while Chapter 5 evaluates the frequency of occurrence. A binary outcome is natural when evaluating the result of a single discrete event as in Chapter 4, while a frequency is more natural when evaluating the risk of continuing the mission with this parameter configuration as in Chapter 5.

The 1.5 DBN approach used in Chapter 5 makes the computational burden insignificant compared to Chapter 4 and Part II where it put limitations on the inference capabilities. The drawback of the 1.5 DBN approach is that it forgets all dependencies of past states. This would be too limiting for Chapter 4 and Part II as the focus of these chapters is to identify the underlying states based on a history of observations. This is also a focus in Chapter 5, but is here limited to differentiate between states that are varying slowly and quickly, which is adequately modeled with the 1.5 DBN approach. Remembering all past states as in Chapter 4 enables the system to evaluate past states with new information which makes it possible to alleviate past mistakes. This ability is not relevant for continuous control as in Chapter 5 and Part II, as it is not possible for these systems to return to a past situation and undo the error.

Each chapter in Part II presents different ways of validating the intention model. In Chapter 6 the model is tested on constructed simulation scenarios. Using simulated scenarios enables the testing of large quantities of scenarios with gradual variations in initial state. This demonstrates how sensitive the model is to the initial state. Chapter 7 tests the model in a real-time setting on controlled experiments. This evaluated how the model holds up to real-time use. Additionally, it evaluates how the model works with real data which has significantly more noise and random variations than the simulated data. Lastly, Chapter 8 tests the model on historical data. This evaluates how well the model works on behavior that actually happens at sea.

The intention model introduced in Chapter 6 considers uncertainty in when the situation started and whether there is a risk of collision. These uncertainties are removed in Chapter 7 as they were deemed irrelevant for the experiments presented in this chapter but are included again in Chapter 8. How “risk of collision” is modeled is shown to work well in Chapter 8. Regarding “situation started” Chapter 8 shows that a way of filtering out maneuvers not related to collision avoidance is important for understanding the intentions of ships, but that the method presented in Chapter 6 for modeling when the situation starts might not be the ideal solution.

9.2 Contribution to research questions, and future work

RQ 1 How can uncertainty in static obstacles be modeled when considering the uncertainty in the sensor mapping the obstacle, and how can this be combined with uncertainty in the navigation of the robotic system itself in a risk-based collision avoidance algorithm?

Chapter 3 contributes towards RQ 1 by making an obstacle avoidance algorithm that uses the probability of collision considering both the obstacle uncertainty and the navigational uncertainty of the drone. The obstacle uncertainty is modeled as an occupancy grid map that is updated considering the uncertainty in the measured distance and where inside the field of view of the sensor the obstacle is. Even though this chapter contributes towards RQ 1, it is still an open question of how to model the uncertainty in the environment in a realistic manner. This chapter uses occupancy grid maps which do not store information on conditional dependencies between cells. This makes the probability value in each cell depend on factors that should be unrelated, such as the grid cell size. The resulting values in each cell can be interpreted as an occupancy level, but can not be interpreted as an approximation of the real probability. Opportunities for future work on RQ 1 are therefore to find a probabilistic sound way to model uncertainty in the environment.

RQ 2 How can the intention of other agents at sea be modeled and inferred in greater detail so that the future behavior of the agent can be predicted with higher accuracy?

Chapter 6 contributes towards RQ 2 by developing an intention model that can be used to infer the intentions of other ships in a collision avoidance situation at sea. The model enables the autonomous ship to identify situations prone to cause misunderstandings and to understand the intention of other ships with greater accuracy than simply being compliant or not. The model manages well to identify when ships do COLREGs incompliant evasive maneuvers, when they act as if they have higher priority, and whether there is a risk of collision. A limitation of the proposed method is that it does not consider grounding hazards and traffic separation schemes. Future work on finding a formulation that takes these cases into account is needed for the intention model to be complete enough to operate in most places.

Chapter 8 contributes further towards RQ 2 by evaluating the intention model on historical data using empirical distributions. This chapter verifies that the model works on real ship behavior. This chapter replaced some of the example distributions used to describe ship behavior with empirical distributions based on the AIS data. Only the distributions that are easily available from the already classified data are used. Further work is needed on finding experimental distributions for the last distributions as well. Furthermore, it was shown that applying the same distribution to all cases was not necessarily an ideal solution. Further work could therefore be done on adapting the distribution to the current situation. Finally, some weaknesses with the model are presented in the chapter, mainly related to identifying when the situation starts. Further work is needed on finding a better way to filter away all maneuvers not related to collision avoidance.

RQ 3 How can an improved situation awareness of the other ships' intentions improve the collision avoidance capabilities of autonomous ships?

Chapter 7 contributes towards RQ 3 by making an intention aware COLAV system which combines the intention model with the PSB-MPC collision avoidance algorithm and experimentally tested the resulting system. The experiments demonstrate that the system is able to give way correctly and to fulfill its stand-on obligations. Furthermore it demonstrates that the system is able to break its stand-on obligations when it becomes apparent that the other ship will not give way, and break its COLREGs compliance obligations when needed to due to the other ship disregards COLREGs. A limitation of the presented experiments is that they are done in confined waters while the algorithm is designed primarily for encounters at open sea. Future work could therefore be done on full-scale experiments in open waters. A limitation of the intention model is that it too easily locks into the unmodeled behavior state. This was often caused by random course changes that had no underlying intention, or by small maneuvers to avoid static obstacles. Future work could be done on alleviating this problem, by for example having a way for the intention model to disregard measurements that turns out to be outliers.

RQ 4 How can a robotic system build and use risk awareness for operational decision-making for considering which task or action to do next?

Chapter 4 contributes towards RQ 4 by making a DDN which increased the risk awareness and decision-making capabilities of a drone. A holistic model is developed that can identify the cause of the underlying problems by combining information on the choice of actions and observed results over time. This enables the system to consider the uncertainty in the system's knowledge about the state of the drone and its environment when making risk-based decisions. Additionally, this chapter demonstrates that evaluating past states with new information can be useful as it enables the system to identify and handle past sub-optimal actions. This information enables the system to return to past tasks which it has wrongly skipped. A limitation of the work presented in this chapter is the computational complexity of keeping all past decisions in the DDN. Opportunities for future work is to find a formulation that requires remembering fewer past states without compromising the inference capabilities of the model too much.

RQ 5 How can a robotic system build and use risk awareness for continuous decisions that have to be made during a task execution?

Chapter 5 contributes towards RQ 5 by making a method for increasing risk awareness for continuous monitoring and control during the mission execution. A holistic model is developed that infers the state of different causal factors by combining information from different measurements over time. The model is then used to make risk-based decisions by considering the different underlying causal facts in light of each other. The experimental trials demonstrate that the system can modify safety-critical parameters in real-time to increase safety. The focus of this chapter is to present a proof of concept of a supervisory risk controller for a manually controlled drone operation. Future developments of supervisory risk control could be done on applying the proposed method to different case studies, such as for a more autonomous operation without direct human supervision, or by considering a larger system where a more complicated model is warranted.

RQ 6 How can a situation awareness model based on the results from a risk analysis be used to ensure safer autonomy?

Both Chapters 4 and 5 contribute towards RQ 6. Chapter 4 considers that the output of a risk analysis should be kept in mind when designing the failure-cause nodes. Without an understanding of how the system can fail it is impossible to decide which failure-causes nodes that are needed. Chapter 5 goes further in this regard by actually performing the risk analysis and more directly using the results of the risk analysis in the modeling phase. Here the loss scenarios identified with the STPA are directly used as part of the DBN. These works thereby contribute towards

RQ 6 by making a situation awareness model based on the results of a risk analysis. Chapter 5 considered only a subset of the loss scenarios identified with the STPA. Further work can therefore be done on making a model that incorporates all of the identified scenarios. Furthermore, it was at times quite challenging to use STPA for making the supervisory risk controller presented in Chapter 5, which warranted the changes proposed throughout Chapter 5. Future work could be done on further improving the risk analysis process for making a supervisory risk controller.

An autonomous robot mission requires both discrete decision-making about which action to do next and continuous evaluation and control of how the actions are executed. Future work could be done on combining the works in Chapters 4 and 5 in a single joint risk-aware autonomous system. The risk-based discrete decision-making system could then evaluate a maximum acceptable risk threshold for executing the task, which the supervisory risk controller would use during the execution of the task. If there is no way of performing the task within the risk threshold, then the supervisory risk controller would make the system return to a safe state and request that the decision-making-system re-plans.

9.3 Concluding remarks

The goal of this thesis was to give robotic systems at least parts of the risk-awareness that skilled human operators possess. This thesis has contributed towards this goal with respect to better modeling of uncertainties in the environment and uncertainties in other agents' intentions, and by developing holistic risk awareness models for discrete and continuous decision making.

Most of the presented works have used BBNs, or their dynamic extensions DBNs and DDNs. BBNs have in recent years been much used in system health management and risk modeling, but have received limited attention for automatic control. This thesis has demonstrated different capabilities that can be achieved by using BBNs for automatic control, especially with respect to improving the situation awareness of the system. Based on the demonstrated capabilities and due to the interpretability of BBNs, I believe that they have potential for making smarter and safer high-level controllers of future autonomous systems.

The intention model developed in this thesis has demonstrated new capabilities that are crucial for understanding the behavior of meeting traffic. The model presented in this thesis can work as a starting point for developing new and better intention models that are more robust and work in a broader set of conditions. I believe that the core concept of using a DBNs in the proposed manner has virtue and should be considered in future intention inference algorithms.

This thesis has presented interdisciplinary work by using theories from risk management in cybernetic systems. Most risk-aware controllers developed in the cybernetics discipline tend to have a limited view of risk by only considering single factors, most often navigation uncertainty. Including the holistic view of risk, which is the focus of risk management, is crucial for ensuring safe systems. This can be done in the traditional way of making system requirements and ensuring that the proposed controller fulfills all requirements, or with the method explored in this thesis of making risk models that are an active part of the control system. Based on the results of this thesis, I believe that using an online risk model as part of the controller will enable us to make safer and more efficient systems, as having a risk model rather than a set of constraints can potentially make the system's behavior less conservative without compromising safety.

References

[1] I. Schjølberg and I. B. Utne, "Towards autonomy in ROV operations," *IFAC-PapersOnLine*, vol. 48, no. 2, pp. 183–188, 2015. DOI: 10.1016/j.ifacol.2015.06.030.

[2] M. Endsley, "Autonomous Horizons: System Autonomy in the Air Force - A Path to the Future," United States Air Force, Washington DC, Tech. Rep. 2015-0267, 2015, p. 27.

[3] A. G. Bruzzone, M. Massei, R. Di Matteo, and L. Kutej, *Introducing Intelligence and Autonomy into Industrial Robots to Address Operations into Dangerous Area*. Springer International Publishing, 2019, vol. 11472 LNCS, pp. 433–444. DOI: 10.1007/978-3-030-14984-0_32.

[4] M. L. Seto, *Marine Robot Autonomy*, M. L. Seto, Ed. New York, NY: Springer New York, 2013. DOI: 10.1007/978-1-4614-5659-9.

[5] C. Wong, E. Yang, X.-T. Yan, and D. Gu, "Autonomous robots for harsh environments: a holistic overview of current solutions and ongoing challenges," *Systems Science & Control Engineering*, vol. 6, no. 1, pp. 213–219, Jan. 2018. DOI: 10.1080/21642583.2018.1477634.

[6] P. Espen, Y. Hegrens, B. Jalving, I. Midtgaard, M. Wiig, and O. Kent, "Making AUVs Truly Autonomous," in *Underwater Vehicles*, A. V. Inzartsev, Ed., Vienna, Austria: InTech, Jan. 2009, ch. ch. 8, pp. 129–152. DOI: 10.5772/6700.

[7] M. R. Endsley, "From Here to Autonomy: Lessons Learned from Human-Automation Research," *Human Factors*, vol. 59, no. 1, pp. 5–27, 2017. DOI: 10.1177/0018720816681350.

[8] S. Hogenboom, B. Rokseth, J. E. Vinnem, and I. B. Utne, "Human reliability and the impact of control function allocation in the design of dynamic positioning systems," *Reliability Engineering & System Safety*, vol. 194, no. August 2017, p. 106 340, Feb. 2020. DOI: 10.1016/j.ress.2018.12.019.

[9] M. R. Endsley, "Situation Awareness in Future Autonomous Vehicles: Beware of the Unexpected," in *Advances in Intelligent Systems and Computing*, May, vol. 824, 2019, pp. 303–309. DOI: 10.1007/978-3-319-96071-5_32.

[10] A. Jónsson, R. A. Morris, and L. Pedersen, "Autonomy in space: Current capabilities and future challenges," *AI Magazine*, vol. 28, no. 4, pp. 27–42, 2007. DOI: 10.1609/aimag.v28i4.2066.

[11] C. R. German, M. V. Jakuba, J. C. Kinsey, J. Partan, S. Suman, A. Belani, and D. R. Yoerger, "A long term vision for long-range ship-free deep ocean operations: Persistent presence through coordination of Autonomous Surface Vehicles and Autonomous Underwater Vehicles," in *2012 IEEE/OES Autonomous Underwater Vehicles (AUV)*, IEEE, Sep. 2012, pp. 1–7. DOI: 10.1109/AUV.2012.6380753.

[12] M. R. Endsley, "Toward a Theory of Situation Awareness in Dynamic Systems," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 37, no. 1, pp. 32–64, Mar. 1995. DOI: 10.1518/001872095779049543.

[13] IMO, *COLREGs - Convention on the International Regulations for Preventing Collisions at Sea*, London, U.K., 1972.

[14] T. A. Johansen, T. Perez, and A. Cristofaro, "Ship Collision Avoidance and COLREGS Compliance Using Simulation-Based Control Behavior Selection With Predictive Hazard Assessment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3407–3422, Dec. 2016. DOI: 10.1109/TITS.2016.2551780.

[15] L. Quan, Z. Zhang, X. Zhong, C. Xu, and F. Gao, "EVA-Planner: Environmental Adaptive Quadrotor Planning," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May, no. Icra, pp. 3751–3757, Nov. 2020. DOI: 10.1109/ICRA48506.2021.9561759.

[16] A. Wang, P. Zhi, W. Zhu, H. Qiu, H. Wang, and W. Wang, "Path Planning of Unmanned Surface Vehicle Based on A Algorithm Optimization Considering the Influence of Risk Factors," in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, IEEE, May 2021, pp. 810–815. DOI: 10.1109/ICPS49255.2021.9468142.

[17] D. Shen, Y. Chen, L. Li, and S. Chien, "Collision-Free Path Planning for Automated Vehicles Risk Assessment via Predictive Occupancy Map," *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 985–991, 2020. DOI: 10.1109/IV47402.2020.9304720.

[18] G. Garimella, M. Sheckells, and M. Kobilarov, "Robust obstacle avoidance for aerial platforms using adaptive model predictive control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5876–5882. DOI: 10.1109/ICRA.2017.7989692.

[19] L. Blackmore, Hui Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *2006 American Control Conference*, IEEE, 2006, 7 pp. DOI: 10.1109/ACC.2006.1656653.

[20] T. Tengesdal, T. A. Johansen, and E. F. Brekke, "Ship Collision Avoidance Utilizing the Cross-Entropy Method for Collision Risk Assessment," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2021. DOI: 10.1109/TITS.2021.3101007.

[21] B. C. Shah, P. vec, I. R. Bertaska, A. J. Sinisterra, W. Klinger, K. von Ellenrieder, M. Dhanak, and S. K. Gupta, "Resolution-adaptive risk-aware trajectory planning for surface vehicles operating in congested civilian traffic," *Autonomous Robots*, vol. 40, no. 7, pp. 1139–1163, Oct. 2016. DOI: 10.1007/s10514-015-9529-x.

[22] A. Jasour, X. Huang, A. Wang, and B. C. Williams, "Fast nonlinear risk assessment for autonomous vehicles using learned conditional probabilistic models of agent futures," *Autonomous Robots*, vol. 46, no. 1, pp. 269–282, Jan. 2022. DOI: 10.1007/s10514-021-10000-1.

[23] K. Cai, C. Wang, S. Song, H. Chen, and M. Q. Meng, "Risk-Aware Path Planning Under Uncertainty in Dynamic Environments," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 101, no. 3, 2021. DOI: 10.1007/s10846-021-01323-3.

[24] J. Lunenburg, R. van de Molengraft, and M. Steinbuch, "A representation method based on the probability of collision for safe robot navigation in domestic environments," *Autonomous Robots*, vol. 42, no. 3, pp. 601–614, Mar. 2018. DOI: 10.1007/s10514-017-9653-x.

[25] Y. Cho, J. J. Kim, and J. J. Kim, "Intent Inference of Ship Collision Avoidance Behavior Under Maritime Traffic Rules," *IEEE Access*, vol. 9, pp. 5598–5608, 2021. DOI: 10.1109/ACCESS.2020.3048717.

[26] J. L. Yepes, I. Hwang, and M. Rotea, "New Algorithms for Aircraft Intent Inference and Trajectory Prediction," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 2, pp. 370–382, Mar. 2007. DOI: 10.2514/1.26750.

[27] J. Hardy and M. Campbell, "Contingency Planning Over Probabilistic Obstacle Predictions for Autonomous Road Vehicles," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 913–929, Aug. 2013. DOI: 10.1109/TRO.2013.2254033.

[28] Y. Chen, F. Zhao, and Y. Lou, "Interactive Model Predictive Control for Robot Navigation in Dense Crowds," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–13, 2021. DOI: 10.1109/TSMC.2020.3048964.

[29] Y. Hashimoto, Y. Gu, L.-T. Hsu, and S. Kamijo, "Probability estimation for pedestrian crossing intention at signalized crosswalks," in *2015 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, IEEE, Nov. 2015, pp. 114–119. DOI: 10.1109/ICVES.2015.7396904.

[30] N. Lefebvre, I. Schjølberg, and I. B. Utne, "Integration of risk in hierarchical path planning of underwater vehicles," *IFAC-PapersOnLine*, vol. 49, no. 23, pp. 226–231, 2016. DOI: 10.1016/j.ifacol.2016.10.347.

[31] X. Hu, B. Pang, F. Dai, and K. H. Low, "Risk Assessment Model for UAV Cost-Effective Path Planning in Urban Environments," *IEEE Access*, vol. 8, pp. 150 162–150 173, 2020. DOI: 10.1109/ACCESS.2020.3016118.

[32] S. Primatesta, A. Rizzo, and A. la Cour-Harbo, “Ground Risk Map for Unmanned Aircraft in Urban Environments,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 97, no. 3-4, pp. 489–509, 2020. DOI: 10.1007/s10846-019-01015-z.

[33] T. Somers and G. A. Hollinger, “Humanrobot planning and learning for marine data collection,” *Autonomous Robots*, vol. 40, no. 7, pp. 1123–1137, 2016. DOI: 10.1007/s10514-015-9502-8.

[34] A. la Cour-Harbo, “Quantifying Risk of Ground Impact Fatalities for Small Unmanned Aircraft,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 93, no. 1-2, pp. 367–384, 2019. DOI: 10.1007/s10846-018-0853-1.

[35] I. B. Utne, B. Rokseth, A. J. Sørensen, and J. E. Vinnem, “Towards supervisory risk control of autonomous ships,” *Reliability Engineering & System Safety*, vol. 196, no. October 2019, p. 106 757, Apr. 2020. DOI: 10.1016/j.ress.2019.106757.

[36] T. Johansen and I. B. Utne, “Supervisory risk control of autonomous surface ships,” *Ocean Engineering*, vol. 251, no. August 2021, p. 111 045, May 2022. DOI: 10.1016/j.oceaneng.2022.111045.

[37] J. E. Bremnes, C. A. Thieme, A. J. Sørensen, I. B. Utne, and P. Norgren, “A Bayesian Approach to Supervisory Risk Control of AUVs Applied to Under-Ice Operations,” *Marine Technology Society Journal*, vol. 54, no. 4, pp. 16–39, Jul. 2020. DOI: 10.4031/MTSJ.54.4.5.

[38] M. Coombes, W.-H. Chen, and P. Render, “Site Selection During Unmanned Aerial System Forced Landings Using Decision-Making Bayesian Networks,” *Journal of Aerospace Information Systems*, vol. 13, no. 12, pp. 491–495, Dec. 2016. DOI: 10.2514/1.I010432.

[39] Y. Qin, Q. Zhang, C. Zhou, and N. Xiong, “A Risk-Based Dynamic Decision-Making Approach for Cybersecurity Protection in Industrial Control Systems,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 10, pp. 3863–3870, Oct. 2020. DOI: 10.1109/TSMC.2018.2861715.

[40] D. Codetta-Raiteri and L. Portinale, “Dynamic Bayesian Networks for Fault Detection, Identification, and Recovery in Autonomous Spacecraft,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 13–24, Jan. 2015. DOI: 10.1109/TSMC.2014.2323212.

[41] B. Cai, Y. Liu, and M. Xie, “A dynamic-bayesian-network-based fault diagnosis methodology considering transient and intermittent faults,” *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 276–285, 2017. DOI: 10.1109/TASE.2016.2574875.

[42] C. Iamsumang, A. Mosleh, and M. Modarres, “Monitoring and learning algorithms for dynamic hybrid Bayesian network in on-line system health management applications,” *Reliability Engineering & System Safety*, vol. 178, no. May, pp. 118–129, Oct. 2018. DOI: 10.1016/j.ress.2018.05.016.

[43] P. Banerjee and G. Gorospe, “Risk Assessment of Obstacle Collision for UAVs under off-nominal conditions,” *Annual Conference of the PHM Society*, vol. 12, no. 1, p. 9, Nov. 2020. DOI: 10.36001/phmconf.2020.v12i1.1194.

[44] G. Hagele and A. Sarkheyli-Hagele, “Situational risk assessment within safety-driven behavior management in the context of UAS,” in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, Sep. 2020, pp. 1407–1415. DOI: 10.1109/ICUAS48674.2020.9214072.

[45] G. Hagele and A. Sarkheyli-Hagele, “Situational Hazard Recognition and Risk Assessment Within Safety-Driven Behavior Management in the Context of Automated Driving,” in *2020 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, IEEE, Aug. 2020, pp. 188–194. DOI: 10.1109/CogSIMA49017.2020.9216183.

[46] S. V. Rothmund and T. A. Johansen, “Risk-Based Obstacle Avoidance in Unknown Environments using Scenario-Based Predictive Control for an Inspection Drone Equipped with Range Finding Sensors,” in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, Jun. 2019, pp. 221–230. DOI: 10.1109/ICUAS.2019.8797803.

[47] S. V. Rothmund, C. A. Thieme, I. B. Utne, and T. A. Johansen, “A Bayesian Approach to Risk-Based Autonomy for a Robotic System Executing a Sequence of Independent Tasks,” *Submitted to Journal of Intelligent & Robotic Systems*, 2022. DOI: 10.36227/techrxiv.14054141.v3.

[48] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Pearson Education, 2014.

[49] S. V. Rothmund, C. A. Thieme, I. B. Utne, and T. A. Johansen, “Supervisory Risk Control with Application to Industrial Drone Inspection,” *Submitted to Autonomous Robots*, 2022. DOI: 10.36227/techrxiv.21287334.

[50] N. G. Leveson and J. P. Thomas, *STPA Handbook*. MA, USA: Cambridge, 2018, p. 188.

[51] S. V. Rothmund, T. Tengesdal, E. F. Brekke, and T. A. Johansen, “Intention modeling and inference for autonomous collision avoidance at sea,” *Ocean Engineering*, vol. 266, p. 113 080, Dec. 2022. DOI: 10.1016/j.oceaneng.2022.113080.

[52] T. Tengesdal, S. V. Rothmund, E. A. Basso, T. A. Johansen, and H. Schmidt-Didlaukies, “Obstacle Intention Awareness in Automatic Collision Avoidance: Full Scale Experiments in Confined Waters,” *Submitted to Field Robotics*, 2022.

[53] T. Tengesdal, T. A. Johansen, T. A. Grande, and S. Blindheim, “Ship Collision Avoidance and Anti Grounding Using Parallelized Cost Evaluation in Probabilistic Scenario-based Model Predictive Control,” *Submitted to IEEE Access*, 2022.

[54] S. V. Rothmund, H. E. Haugen, G. D. Veglo, E. F. Brekke, and T. A. Johansen, “Validation of ship intention model for maritime collision avoidance control using historical AIS data,” *Submitted to ECC*, 2023.

[55] IMO, *AIS transponders.*

[56] K. B. Korb and A. E. Nicholson, *Bayesian artificial intelligence*, Boca Raton, Fla, 2011.

[57] G. F. Cooper, "The computational complexity of probabilistic inference using bayesian belief networks," *Artificial Intelligence*, vol. 42, no. 2-3, pp. 393–405, Mar. 1990. DOI: 10.1016/0004-3702(90)90060-D.

[58] S. L. Lauritzen and D. J. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 50, no. 2, pp. 157–194, 1988. DOI: 10.1111/j.2517-6161.1988.tb01721.x.

[59] C. Yuan and M. J. Druzdzel, "An Importance Sampling Algorithm Based on Evidence Pre-propagation," *Uncertainty in Artificial Intelligence 19*, pp. 624–631, 2003.

[60] BAYESFUSION LLC, *SMILE Engine.*

[61] Team AGrUM, *aGrUM.*

[62] A. Ben Mrad, V. Delcroix, M. A. Maalej, S. Piechowiak, and M. Abid, "Uncertain Evidence in Bayesian Networks: Presentation and Comparison on a Simple Example," in *Communications in Computer and Information Science*, vol. 299, PART, 2012, pp. 39–48. DOI: 10.1007/978-3-642-31718-7_5.

[63] N. Fenton and M. Neil, *Risk Assessment and Decision Analysis with Bayesian Networks*, 2nd editio. Chapman & Hall/CRC, 2018.

[64] S. Kaplan and B. J. Garrick, "On The Quantitative Definition of Risk," *Risk Analysis*, vol. 1, no. 1, pp. 11–27, Mar. 1981. DOI: 10.1111/j.1539-6924.1981.tb01350.x.

[65] M. J. Kochenderfer, *Decision Making Under Uncertainty.* The MIT Press, 2015. DOI: 10.7551/mitpress/10187.001.0001.

[66] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering, Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, 1960. DOI: 10.1115/1.3662552.

[67] P. Del Moral, "Nonlinear filtering: Interacting particle resolution," *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, vol. 325, no. 6, pp. 653–658, Sep. 1997. DOI: 10.1016/S0764-4442(97)84778-7.

[68] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa, "Online Planning Algorithms for POMDPs," *Journal of Artificial Intelligence Research*, vol. 32, pp. 663–704, Jul. 2008. DOI: 10.1613/jair.2567.

[69] D. Mayne, "Robust and stochastic model predictive control: Are we going in the right direction?" *Annual Reviews in Control*, vol. 41, pp. 184–192, 2016. DOI: 10.1016/j.arcontrol.2016.04.006.

[70] M. Farina, L. Giulioni, and R. Scattolini, "Stochastic linear Model Predictive Control with chance constraints A review," *Journal of Process Control*, vol. 44, pp. 53–67, Aug. 2016. DOI: 10.1016/j.jprocont.2016.03.005.

[71] T. Tengesdal, T. A. Johansen, and E. F. Brekke, “Risk-based Autonomous Maritime Collision Avoidance Considering Obstacle Intentions,” in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, IEEE, Jul. 2020, pp. 1–8. DOI: 10.23919/FUSION45008.2020.9190212.

[72] A. Elfes, “Occupancy Grids: A Stochastic Spatial Representation for Active Robot Perception,” in *6th Conference on Uncertainty in AI*, 1990.

[73] D. Q. Mayne, M. M. Seron, and S. V. Rakovi, “Robust model predictive control of constrained linear systems with bounded disturbances,” *Automatica*, vol. 41, no. 2, pp. 219–224, 2005. DOI: 10.1016/j.automatica.2004.08.019.

[74] A. Majumdar and R. Tedrake, “Funnel libraries for real-time robust feedback motion planning,” *International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017. DOI: 10.1177/0278364917712421.

[75] E. Kaufman, K. Takami, T. Lee, and Z. Ai, “Autonomous Exploration with Exact Inverse Sensor Models,” *Journal of Intelligent & Robotic Systems*, vol. 92, no. 3, pp. 435–452, 2018. DOI: 10.1007/s10846-017-0710-7.

[76] J. Borenstein and Y. Koren, “The vector field histogram-fast obstacle avoidance for mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278–288, 1991. DOI: 10.1109/70.88137.

[77] K. Ok, S. Ansari, B. Gallagher, W. Sica, F. Dellaert, and M. Stilman, “Path planning with uncertainty: Voronoi Uncertainty Fields,” in *Proceedings - IEEE International Conference on Robotics and Automation*, IEEE, 2013, pp. 4596–4601. DOI: 10.1109/ICRA.2013.6631230.

[78] Y. Koren and J. Borenstein, “Potential field methods and their inherent limitations for mobile robot navigation,” in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, Apr. 2002, pp. 1398–1404. DOI: 10.1109/robot.1991.131810.

[79] G. V. Pelizer, N. B. Da Silva, and K. R. Branco, “Comparison of 3D path-following algorithms for unmanned aerial vehicles,” in *2017 International Conference on Unmanned Aircraft Systems, ICUAS 2017*, 2017, pp. 498–505. DOI: 10.1109/ICUAS.2017.7991338.

[80] T. I. Fossen, *Handbook of Marine Craft Hydrodynamics and Motion Control.* Wiley, 2011. DOI: 10.1002/9781119994138.

[81] K. Konolige, “Improved Occupancy Grids for Map Building,” *Auton. Robots*, vol. 4, no. 4, pp. 351–367, 1997. DOI: 10.1023/A:1008806422571.

[82] R. Dia, J. Mottin, T. Rakotovao, D. Puschini, and S. Lesecq, “Evaluation of Occupancy Grid Resolution through a Novel Approach for Inverse Sensor Modeling,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 13 841–13 847, 2017. DOI: 10.1016/j.ifacol.2017.08.2225.

[83] I. B. Utne, A. J. Sørensen, and I. Schjølberg, “Risk Management of Autonomous Marine Systems and Operations,” in *Volume 3B: Structures, Safety and Reliability*, Trondheim: American Society of Mechanical Engineers, Jun. 2017. DOI: 10.1115/OMAE2017-61645.

[84] ISO 31000:2018, "Risk management - principles and guidelines," International Organization for Standardization, Geneva, Switzerland, Standard, 2018.

[85] K. Rajan and A. Saffiotti, "Towards a science of integrated AI and Robotics," *Artificial Intelligence*, vol. 247, pp. 1–9, Jun. 2017. DOI: 10.1016/j.artint.2017.03.003.

[86] J. Luque and D. Straub, "Reliability analysis and updating of deteriorating systems with dynamic Bayesian networks," *Structural Safety*, vol. 62, pp. 34–46, 2016. DOI: 10.1016/j.strusafe.2016.03.004.

[87] I. P. Gomes and D. F. Wolf, "Health Monitoring System for Autonomous Vehicles using Dynamic Bayesian Networks for Diagnosis and Prognosis," *Journal of Intelligent & Robotic Systems*, vol. 101, no. 1, p. 19, Jan. 2021. DOI: 10.1007/s10846-020-01293-y.

[88] Y. Hernández, J. Noguez, E. Sucar, and G. Arroyo-Figueroa, "Incorporating an affective model to an intelligent tutor for mobile robotics," *Proceedings - Frontiers in Education Conference, FIE*, pp. 22–27, 2006. DOI: 10.1109/FIE.2006.322407.

[89] R. Charles Murray, K. VanLehn, and J. Mostow, "Looking ahead to select tutorial actions: A decision-theoretic approach," *International Journal of Artificial Intelligence in Education*, vol. 14, no. 3-4, pp. 235–278, 2004.

[90] C. Conati, "Probabilistic assessment of user's emotions in educational games," *Applied Artificial Intelligence*, vol. 16, no. 7-8, pp. 555–575, 2002. DOI: 10.1080/08839510290030390.

[91] B. W. Mott and J. C. Lester, "U-DIRECTOR: A decision-theoretic narrative planning architecture for storytelling environments," *Proceedings of the International Conference on Autonomous Agents*, vol. 2006, pp. 977–984, 2006. DOI: 10.1145/1160633.1160808.

[92] T. H. Bui, M. Poel, A. Nijholt, and J. Zwiers, "A tractable hybrid DDN-POMDP approach to affective dialogue modeling for probabilistic frame-based dialogue systems," *Natural Language Engineering*, vol. 15, no. 2, pp. 273–307, 2009. DOI: 10.1017/S1351324908005032.

[93] M. Trujillo, J. Martínez-de Dios, C. Martín, A. Viguria, and A. Ollero, "Novel Aerial Manipulator for Accurate and Robust Industrial NDT Contact Inspection: A New Tool for the Oil and Gas Inspection Industry," *Sensors*, vol. 19, no. 6, p. 1305, Mar. 2019. DOI: 10.3390/s19061305.

[94] R. A. Mattar and R. Kalai, "Development of a wall-sticking drone for non-destructive ultrasonic and corrosion testing," *Drones*, vol. 2, no. 1, pp. 1–11, 2018. DOI: 10.3390/drones2010008.

[95] B. B. Kocer, T. Tjahjowidodo, M. Pratama, and G. G. L. Seet, "Inspection-while-flying: An autonomous contact-based nondestructive test using UAV-tools," *Automation in Construction*, vol. 106, no. July, p. 102 895, 2019. DOI: 10.1016/j.autcon.2019.102895.

[96] L. M. González-deSantos, J. Martínez-Sánchez, H. González-Jorge, F. Navarro-Medina, and P. Arias, "UAV payload with collision mitigation for contact inspection," *Automation in Construction*, vol. 115, no. March, p. 103 200, 2020. DOI: 10.1016/j.autcon.2020.103200.

[97] D. Zhang, R. Watson, G. Dobie, C. MacLeod, and G. Pierce, "Autonomous Ultrasonic Inspection Using Unmanned Aerial Vehicle," *IEEE International Ultrasonics Symposium, IUS*, vol. 2018-Octob, 2018. DOI: 10.1109/ULTSYM.2018.8579727.

[98] G. Shani, "Task-based decomposition of factored POMDPs," *IEEE Transactions on Cybernetics*, vol. 44, no. 2, pp. 208–216, 2014. DOI: 10.1109/TCYB.2013.2252009.

[99] M. Rausand and S. Haugen, *Risk Assessment*, 1st ed. New Jersey: Wiley, Mar. 2020. DOI: 10.1002/9781119377351.

[100] L. Mkrtchyan, L. Podofillini, and V. Dang, "Methods for building Conditional Probability Tables of Bayesian Belief Networks from limited judgment: An evaluation for Human Reliability Application," *Reliability Engineering & System Safety*, vol. 151, pp. 93–112, Jul. 2016. DOI: 10.1016/j.ress.2016.01.004.

[101] DNV GL AS, "Non-destructive testing," Tech. Rep. December, 2015.

[102] A. A. Pereira, J. Binney, G. A. Hollinger, and G. S. Sukhatme, "Risk-aware Path Planning for Autonomous Underwater Vehicles using Predictive Ocean Models," *Journal of Field Robotics*, vol. 30, no. 5, pp. 741–762, Sep. 2013. DOI: 10.1002/rob.21472.

[103] J. Pearl, "Bayesian networks: A model of self-activated memory for evidential reasoning," in *Proceedings of the 7th Conference of the Cognitive Science Society, 1985*, 1985, pp. 329–334.

[104] C. A. Thieme, B. Rokseth, and I. B. Utne, "Risk-informed control systems for improved operational performance and decision-making," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, Aug. 2021. DOI: 10.1177/1748006X211043657.

[105] A. Allouch, A. Koubaa, M. Khalgui, and T. Abbes, "Qualitative and Quantitative Risk Analysis and Safety Assessment of Unmanned Aerial Vehicles Missions Over the Internet," *IEEE Access*, vol. 7, pp. 53 392–53 410, 2019. DOI: 10.1109/ACCESS.2019.2911980.

[106] L. C. Barr, R. Newman, E. Ancel, C. M. Belcastro, J. V. Foster, J. Evans, and D. H. Klyde, "Preliminary Risk Assessment for Small Unmanned Aircraft Systems," in *17th AIAA Aviation Technology, Integration, and Operations Conference*, Reston, Virginia: American Institute of Aeronautics and Astronautics, Jun. 2017. DOI: 10.2514/6.2017-3272.

[107] A. Plioutsias, N. Karanikas, and M. M. Chatzimihailidou, "Hazard Analysis and Safety Requirements for Small Drone Operations: To What Extent Do Popular Drones Embed Safety?" *Risk Analysis*, vol. 38, no. 3, pp. 562–584, Mar. 2018. DOI: 10.1111/risa.12867.

[108] M. M. Chatzimichailidou, N. Karanikas, and A. Plioutsias, “Application of STPA on Small Drone Operations: A Benchmarking Approach,” *Procedia Engineering*, vol. 179, pp. 13–22, 2017. DOI: 10.1016/j.proeng.2017.03.091.

[109] M. Aliyari, B. Ashrafi, and Y. Z. Ayele, “Hazards identification and risk assessment for UAVassisted bridge inspections,” *Structure and Infrastructure Engineering*, vol. 18, no. 3, pp. 412–428, 2022. DOI: 10.1080/15732479.2020.1858878.

[110] A. A. Tubis, J. Ryczyski, and A. urek, “Risk Assessment for the Use of Drones in Warehouse Operations in the First Phase of Introducing the Service to the Market,” *Sensors*, vol. 21, no. 20, p. 6713, Oct. 2021. DOI: 10.3390/s21206713.

[111] F. U. Muram and M. Atif Javed, “Drone-based Risk Management of Autonomous Systems Using Contracts and Blockchain,” in *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, IEEE, Mar. 2021, pp. 679–688. DOI: 10.1109/SANER50967.2021.00086.

[112] S. Charalampidou, E. Lygouras, I. Dokas, A. Gasteratos, and A. Zacharopoulou, “A Sociotechnical Approach to UAV Safety for Search and Rescue Missions,” in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, IEEE, Sep. 2020, pp. 1416–1424. DOI: 10.1109/ICUAS48674.2020.9213921.

[113] Y. Z. Ayele and B. Ashraf, “Preliminary hazard analysis for UAV-assisted bridge inspection,” in *WIT Transactions on the Built Environment*, vol. 200, Nov. 2020, pp. 171–184. DOI: 10.2495/UT200141.

[114] J. T. Luxhøj, “A Socio-technical Model for Analyzing Safety Risk of Unmanned Aircraft Systems (UAS): An Application to Precision Agriculture,” *Procedia Manufacturing*, vol. 3, no. Ahfe, pp. 928–935, 2015. DOI: 10.1016/j.promfg.2015.07.140.

[115] X. Wang, Y. Zhang, L. Wang, J. Wang, and J. Lu, “Maintenance grouping optimization with system multi-level information based on BN lifetime prediction model,” *Journal of Manufacturing Systems*, vol. 50, no. October 2018, pp. 201–211, Jan. 2019. DOI: 10.1016/j.jmsy.2019.01.002.

[116] A. Khayyati and M. Pourgol-Mohammad, “Developing an Efficient Approach for Unmanned Aerial Vehicle Reliability Analysis,” in *Volume 14: Safety Engineering, Risk, and Reliability Analysis*, vol. 14, American Society of Mechanical Engineers, Nov. 2020, pp. 1–9. DOI: 10.1115/IMECE2020-24079.

[117] N. U. I. Hossain, N. Sakib, and K. Govindan, “Assessing the performance of unmanned aerial vehicle for logistics and transportation leveraging the Bayesian network approach,” *Expert Systems with Applications*, vol. 209, no. July, p. 118 301, Dec. 2022. DOI: 10.1016/j.eswa.2022.118301.

[118] ASTM F3269-21, "Standard practice for methods to safely bound behavior of aircraft systems containing complex functions using run-time assurance," ASTM International, USA, Standard, 2021.

[119] B. Rokseth, I. B. Utne, and J. E. Vinnem, "A systems approach to risk analysis of maritime operations," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 231, no. 1, pp. 53–68, Feb. 2017. DOI: 10.1177/1748006X16682606.

[120] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science, 220(4598)*, vol. 220, no. 4598, pp. 671–680, 1983.

[121] S. Renooij, "Probability elicitation for belief networks: issues to consider," *The Knowledge Engineering Review*, vol. 16, no. 3, pp. 255–269, Sep. 2001. DOI: 10.1017/S0269888901000145.

[122] L. Mkrtchyan, L. Podofillini, and V. N. Dang, "Operational and organizational barriers as means of enhancing safety and learning," in *Safety and Reliability: Methodology and Applications*, CRC Press, Sep. 2014, pp. 1083–1092. DOI: 10.1201/b17399-151.

[123] N. E. Fenton, M. Neil, and J. G. Caballero, "Using Ranked Nodes to Model Qualitative Judgments in Bayesian Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 10, pp. 1420–1432, Oct. 2007. DOI: 10.1109/TKDE.2007.1073.

[124] M. S. Hamada, A. G. Wilson, C. S. Reese, and H. F. Martz, *Bayesian Reliability* (Springer Series in Statistics). New York, NY: Springer New York, 2008. DOI: 10.1007/978-0-387-77950-8.

[125] BAYESFUSION LLC, *GeNIe*.

[126] K. P. Murphy, "Dynamic Bayesian Networks," in *Unpublished chapter*, 2002.

[127] B. Rokseth, I. B. Utne, and J. E. Vinnem, "Deriving verification objectives and scenarios for maritime systems using the systems-theoretic process analysis," *Reliability Engineering & System Safety*, vol. 169, no. July 2017, pp. 18–31, Jan. 2018. DOI: 10.1016/j.ress.2017.07.015.

[128] B. Rokseth and I. B. Utne, "Deriving Safety Requirement Hierarchies for Families of Maritime Systems," *International Journal of Maritime Engineering*, vol. 161, no. A3, A229–A243, Sep. 2019. DOI: 10.3940/rina.ijme.2019.a3.526.

[129] C. Chauvin, "Human Factors and Maritime Safety," *Journal of Navigation*, vol. 64, no. 4, pp. 625–632, Oct. 2011. DOI: 10.1017/S0373463311000142.

[130] C. Chauvin and S. Lardjane, "Decision making and strategies in an interaction situation: Collision avoidance at sea," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 11, no. 4, pp. 259–269, 2008. DOI: 10.1016/j.trf.2008.01.001.

[131] S. R. Clawson Jr. "Overtaking or crossing? Don't assume what other ship will do." (2013), [Online]. Available: http://www.professionalmariner.com/August-2013/Overtaking-or-crossing-Dont-assume-what-other-ship-will-do/ (visited on 10/06/2022).

[132] K. Woerner, M. R. Benjamin, M. Novitzky, and J. J. Leonard, "Quantifying protocol evaluation for autonomous collision avoidance," *Autonomous Robots*, vol. 43, no. 4, pp. 967–991, Apr. 2019. DOI: 10.1007/s10514-018-9765-y.

[133] Y. Huang, L. Chen, P. Chen, R. R. Negenborn, and P. van Gelder, "Ship collision avoidance methods: State-of-the-art," *Safety Science*, vol. 121, no. April 2019, pp. 451–473, Jan. 2020. DOI: 10.1016/j.ssci.2019.09.018.

[134] A. Vagale, R. Oucheikh, R. T. Bye, O. L. Osen, and T. I. Fossen, "Path planning and collision avoidance for autonomous surface vehicles I: a review," *Journal of Marine Science and Technology*, vol. 26, no. 4, pp. 1292–1306, Dec. 2021. DOI: 10.1007/s00773-020-00787-6.

[135] B.-O. H. Eriksen, G. Bitar, M. Breivik, and A. M. Lekkas, "Hybrid Collision Avoidance for ASVs Compliant With COLREGs Rules 8 and 1317," *Frontiers in Robotics and AI*, vol. 7, no. February, pp. 1–18, Feb. 2020. DOI: 10.3389/frobt.2020.00011.

[136] L. Du, F. Goerlandt, O. A. Valdez Banda, Y. Huang, Y. Wen, and P. Kujala, "Improving stand-on ship's situational awareness by estimating the intention of the give-way ship," *Ocean Engineering*, vol. 201, no. February, p. 107 110, 2020. DOI: 10.1016/j.oceaneng.2020.107110.

[137] J. Krozel and D. Andrisani, "Intent Inference with Path Prediction," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 2, pp. 225–236, Mar. 2006. DOI: 10.2514/1.14348.

[138] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*, Third Edit. Berlin, Heidelberg: Springer, 2016. DOI: 10.1007/978-3-662-47943-8.

[139] E. T. Jaynes, *Probability Theory - The Logic of Science.* Cambridge University Press, 2003.

[140] J. Hegde, I. B. Utne, I. Schjølberg, and B. Thorkildsen, "A Bayesian approach to risk modeling of autonomous subsea intervention operations," *Reliability Engineering and System Safety*, vol. 175, no. February, pp. 142–159, 2018. DOI: 10.1016/j.ress.2018.03.019.

[141] IMO. "AIS transponders." (n.d.), [Online]. Available: https://www.imo.org/en/OurWork/Safety/Pages/AIS.aspx (visited on 05/19/2021).

[142] A. G. Loe, "Collision Avoidance for Unmanned Surface Vehicles," Ph.D. dissertation, NTNU, 2008.

[143] G. Casalino, A. Turetta, and E. Simetti, "A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields," in *OCEANS 2009-EUROPE*, 2009, pp. 1–8. DOI: 10.1109/OCEANSE.2009.5278104.

[144] T. Trym, E. F. Brekke, and T. A. Johansen, "On Collision Risk Assessment for Autonomous Ships Using Scenario-Based MPC," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14 509–14 516, 2020. DOI: 10.1016/j.ifacol.2020.12.1454.

[145] C. Tam, R. Bucknall, and A. Greig, "Review of Collision Avoidance and Path Planning Methods for Ships in Close Range Encounters," *Journal of Navigation*, vol. 62, no. 3, pp. 455–476, Jul. 2009. DOI: 10.1017/S0373463308005134.

[146] S. Campbell, W. Naeem, and G. W. Irwin, "A review on improving the autonomy of unmanned surface vehicles through intelligent collision avoidance manoeuvres," *Annual Reviews in Control*, vol. 36, no. 2, pp. 267–283, 2012. DOI: https://doi.org/10.1016/j.arcontrol.2012.09.008.

[147] P. Chen, Y. Huang, J. Mou, and P. van Gelder, "Probabilistic risk analysis for ship-ship collision: State-of-the-art," *Safety Science*, vol. 117, pp. 108–122, Aug. 2019. DOI: 10.1016/j.ssci.2019.04.014.

[148] A. Vagale, R. T. Bye, R. Oucheikh, O. L. Osen, and T. I. Fossen, "Path planning and collision avoidance for autonomous surface vehicles II: a comparative study of algorithms," *Journal of Marine Science and Technology*, 2021. DOI: 10.1007/s00773-020-00790-x.

[149] M. Akda\ug, P. Solnør, and T. A. Johansen, "Collaborative collision avoidance for Maritime Autonomous Surface Ships: A review," *Ocean Engineering*, vol. 250, p. 110 920, 2022.

[150] P. Agrawal and J. M. Dolan, "COLREGS-compliant target following for an Unmanned Surface Vehicle in dynamic environments," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 1065–1070. DOI: 10.1109/IROS.2015.7353502.

[151] M. Candeloro, A. M. Lekkas, and A. J. Sørensen, "A Voronoi-diagram-based dynamic path-planning system for underactuated marine vessels," *Control Engineering Practice*, vol. 61, pp. 41–54, Apr. 2017. DOI: 10.1016/j.conengprac.2017.01.007.

[152] X. Rong Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, 2003. DOI: 10.1109/TAES.2003.1261132.

[153] L. M. Millefiori, P. Braca, K. Bryan, and P. Willett, "Long-term vessel kinematics prediction exploiting mean-reverting processes," in *FUSION 2016 - 19th International Conference on Information Fusion, Proceedings*, 2016, pp. 232–239.

[154] H.-T. L. Chiang and L. Tapia, "COLREG-RRT: An RRT-Based COLREGS-Compliant Motion Planner for Surface Vehicle Navigation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2024–2031, Jul. 2018. DOI: 10.1109/LRA.2018.2801881.

[155] Y. Kuwata, M. T. Wolf, D. Zarzhitsky, and T. L. Huntsberger, "Safe maritime autonomous navigation with COLREGS, using velocity obstacles," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 110–119, 2014. DOI: 10.1109/JOE.2013.2254214.

[156] B. Kluge and E. Prassler, "Recursive Probabilistic Velocity Obstacles for Reflective Navigation," in vol. 24, 2006, pp. 71–79. DOI: 10.1007/10991459_8.

[157] D. K. M. Kufoalor, E. F. Brekke, and T. A. Johansen, “Proactive Collision Avoidance for ASVs using A Dynamic Reciprocal Velocity Obstacles Method,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018, pp. 2402–2409. DOI: 10.1109/IROS.2018.8594382.

[158] Y. Cho, J. Han, and J. Kim, “Efficient COLREG-Compliant Collision Avoidance in Multi-Ship Encounter Situations,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2020. DOI: 10.1109/tits.2020.3029279.

[159] Y. Cho, J. Kim, and J. Kim, “Intent Inference-Based Ship Collision Avoidance in Encounters With Rule-Violating Vessels,” *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 518–525, 2022. DOI: 10.1109/LRA.2021.3130386.

[160] Y. Cho, J. Han, and J. Kim, “Intent inference of ship maneuvering for automatic ship collision avoidance,” *IFAC-PapersOnLine*, vol. 51, no. 29, pp. 384–388, 2018. DOI: 10.1016/j.ifacol.2018.09.457.

[161] Y. Cho and J. Kim, “Collision probability assessment between surface ships considering maneuver intentions,” in *OCEANS 2017 - Aberdeen*, vol. 2017-Octob, IEEE, Jun. 2017, pp. 1–5. DOI: 10.1109/OCEANSE.2017.8084791.

[162] S. R. Clawson Jr, *Overtaking or crossing? Dont assume what other ship will do*, 2013.

[163] J. A. García Maza and R. P. Argüelles, “Colregs and their application in collision avoidance algorithms: A critical analysis,” *Ocean Engineering*, vol. 261, p. 112 029, 2022. DOI: https://doi.org/10.1016/j.oceaneng.2022.112029.

[164] Y. Bar-Shalom and X.-R. Li, *Multitarget-multisensor tracking: principles and techniques*. YBs Storrs, CT, 1995, vol. 19.

[165] P. Solnør, Ø. Volden, K. Gryte, S. Petrovic, and T. I. Fossen, “Hijacking of unmanned surface vehicles: A demonstration of attacks and countermeasures in the field,” *Journal of Field Robotics*, vol. 39, no. 5, pp. 631–649, Aug. 2022. DOI: 10.1002/rob.22068.

[166] J. Rodriguez, M. P. Kazmierkowski, J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. A. Young, and C. A. Rojas, “State of the Art of Finite Control Set Model Predictive Control in Power Electronics,” *IEEE Transactions on Industrial Informatics*, vol. 9, no. 2, pp. 1003–1016, May 2013. DOI: 10.1109/TII.2012.2221469.

[167] P. Karamanakos, T. Geyer, N. Oikonomou, F. D. Kieferndorf, and S. Manias, “Direct Model Predictive Control: A Review of Strategies That Achieve Long Prediction Intervals for Power Electronics,” *IEEE Industrial Electronics Magazine*, vol. 8, no. 1, pp. 32–43, Mar. 2014. DOI: 10.1109/MIE.2013.2290474.

[168] K. Granström, M. Baum, and S. Reuter, “Extended OBJECT TRACKING: Introduction, overview, and applications,” *Journal of Advances in Information Fusion*, vol. 12, no. 2, pp. 139–174, 2017.

[169] I. B. Hagen, “Topics on Marine Collision Avoidance,” Ph.D. dissertation, Norwegian University of Science and Technology, 2022.

[170] O. B. Vassbotn, “Analysis of ship collision risk encounters and COLREG behaviours using machine learning and AIS data,” Master’s thesis, Norwegian University of Science and Technology, Jan. 2022.

[171] M. N. Murvold, “Evaluation of COLREGs compliance when considering grounding hazards,” Master’s thesis, Norwegian University of Science and Technology, Jun. 2022.

[172] K. S. Knutsen, “Analysis of AIS data for COLREGS compliance,” Master’s thesis, Norwegian University of Science and Technology, Jun. 2022.