Qualitative Process Analysis

Theoretical Requirements and Practical Implementation in Naval Domain

Immo Colonius

Dissertation
zur Erlangung des Grades eines Doktors der
Ingenieurwissenschaften

— Dr. Ing. —

Vorgelegt im Fachbereich 3 (Mathematik & Informatik)
Universität Bremen
Mai 2015

Datum des Promotionskolloquiums:

Gutachter: Prof. Christian Freksa, Ph.D. (Universität Bremen)
Prof. Dr. Maren Bennewitz (Universität Bonn)

Acknowledgements

The presented work was written while being a doctoral student and researcher in the Cognitive Systems work group of the University of Bremen. I am very grateful for all the support and discussions that I had within this excellent work environment. Especially thankful I am towards my advisor Professor Christian Freksa for the many interesting suggestions and encouragements. His continued constructive discussions with me were essential for this work.

While Christian Freksa provided me with many new viewpoints, Lutz Frommberger and Carl Schultz brought me back on track whenever I deviated from the bigger picture. Especially, Carl's continued guidance and optimism helped me to survive this adventure.

Speaking of essential elements of this work, the productive environment of the R3-Project is (still) held in high regard by me. Thank you, Arne, Jae, Frank, Diedrich and Lutz for sharing words and coffee, frustration and achievements, as well as darts and so much robotic tinkering with me.

Doubtless, more colleagues were significant for this work, I give my heartfelt thanks to Rasmus Wienemann for his friendship and graphical skills, Michaela Bunke for a better idea how to manage workloads and Christian Schunke for discussion of invaluable details with me. Thomas Barkowsky's belief in me boosted my own. A big thank you also has to go towards my friend Johannes Höefer for exemplifying that working hard leads to something better.

Helmsman Christian Podebry opened me a window into the realm of sea navigation and corrected my assumptions about the train of thought of naval officers. In the same track I give my thanks towards the Team of AISHub and their company "Astra Paging Ltd" for providing me with data about ship movements.

My parents Eva Weichert und Fritz Colonius have done more good for me than I can name. They have shown me the value of education and understanding. Your ongoing support made me what I am today. Thank you for being who you are.

Katie, my dear wife, your help, your presence and your energy improves my life so

much. You listened to my thoughts, soothed my worries and fortified my strong points. Thank you for all the time you are spending with me.

Last but not least, Luke, thank you for your ideas and queries that showed me what the really important questions of life are. Kjell, thank you for showing me how splendid parenthood can be and for sharing the nicest achievements with me. As you both show me on a daily basis how wonderful new knowledge can be, I dedicate my thesis to you.

Summary

Understanding complex behaviours is an essential component of everyday life, integrated into daily routines as well as specialised research. To handle the increasing amount of data available from (logistic) dynamic scenarios, analysis of the behaviour of agents in a given environment is becoming more automated and thus requires reliable new analytical methods. This thesis seeks to improve analysis of observed data in dynamic scenarios by developing a new model for transforming sparse behavioural observations into realistic explanations of agent behaviours, with the goal of testing that model in a real-world maritime navigation scenario.

Logistics and operations specialists are concerned with optimising the activities of agents (e.g., to reduce travel time, to minimise costs) and improving safety (e.g., to avoid risk of collision). Examples include maritime navigation, air traffic control, warehouse operations management and cargo transport.

To that end, two main components are central: agents that have movements and activities (i.e., agent behaviours) and the collected data about these behaviours (i.e. sensor data or observations about agents).

However, there are two key challenges. First, observational data gathered by various sensor devices are often noisy and incomplete. Second, descriptions of agent behaviours are typically vague and ambiguous, and may only be available in natural language (i.e., they are informal descriptions), thus preventing rigorous analysis.

Currently, numerical methods are the most commonly used tools for analysing scenarios in operations research. Numerical methods include dynamical systems, in particular, difference and differential equations, statistical models and optimization. These gather numerical values observed from the domain and develop mathematical models (i.e., numerical formulas, differential equations, etc.) that fit the observed data.

However, numerical models are not the most effective approach for modelling intelligent agent activities and behaviours, and therefore only provide limited support for optimisation and safety analysis. First, it is difficult to convert natural language descrip-

tions of agent activities into a system of purely numerical equations, as these so-called expert descriptions tend to be qualitative rather than numerical. Second, generating these mathematical models is a time-consuming, tedious, and error-prone process. Moreover, numerical models are sensitive to data outliers and fail when the data is overly noisy or incomplete.

To address these limitations, I have developed a Knowledge Representation and Reasoning (KR)-based framework for analysing agent behaviour data based on symbolic qualitative models of the domain, i.e., agent movements and activities.

The key mechanism within my framework is an algorithm for matching sensor data about agents with the KR-based descriptions of agent behaviours. Based on this key mechanism for matching, I have developed a range of analytical tools for investigating agent behaviour in a high-level way, which enables optimisation and safety analysis of operations research tasks that involve intelligent agents. Thus, my framework addresses the limitations of a purely numerical modelling approach.

I evaluate my framework with a detailed case study in the domain of maritime navigation. I show that my framework is capable of handling real-world noisy sensor data, can scale to real-world quantities of sensor data, and is sufficiently expressive to formalise real natural language maritime collision avoidance regulations.

Zusammenfassung

In logistischen, dynamischen Szenarien wird für die Verarbeitung der extrem anwachsenden Menge an Daten die Analyse von Verhalten von Agenten in gegebener Umgebung zunehmend automatisiert. Dies erfordert zuverlässige neue Analyse-Methoden. Diese Arbeit zielt darauf ab, die Analyse von beobachteten Daten aus dynamischen Szenarien zu verbessern, indem ein neues Modell für die Übersetzung von spärlichen Beobachtungen in realistische Erklärungen des Agentenverhaltens vorgeschlagen wird und an einem realitätsnahen, maritimen Szenario getestet wird.

Spezialisten aus Logistik und Operations Research haben einerseits das Ziel, die Abläufe ihrer Akteure zu optimieren (beispielsweise Kosten zu senken, Zeiten zu minimieren) und andererseits die Sicherheit der Abläufe zu erhöhen (zum Beispiel Kollisionsrisiken zu vermeiden). Beispielhafte Domänen sind die Seefahrt, die Luftraum-Kontrolle, das Warenhausmanagement oder auch der Gütertransport.

Dabei sind zwei Komponenten zentral: Agenten, die Bewegungen und Aktivitäten aufweisen (Agentenverhalten), sowie Beobachtungs-Daten über diese Bewegungen und Aktivitäten (Beobachtungen). Diese beiden Komponenten bringen jeweils eigene Herausforderungen im Umgang mit. Erstens können Beobachtungsdaten lückenhaft oder verrauscht sein, zweitens sind die Beschreibungen des Agentenverhaltens, die in den Domänen verfügbar sind, oft in natürlich-sprachlicher Form und damit nicht in einer Form, die direkt formalisierbar ist. Dies beeinträchtigt eine automatisierte, präzise Analyse.

Aktuell sind mathematische Modellierungs-Methoden das übliche Mittel, um Analysen in der Logistik oder im Bereich des Operations Research durchzuführen. Dazu zählen dynamische Systeme, insbesondere Differenzen- und Differenzialgleichungen, sowie statistische Methoden und Optimierung. Diese sammeln numerische Werte in der Domäne und entwickeln mathematische Modelle, welche an die beobachteten Daten angepasst werden.

Jedoch sind numerische Verfahren nur begrenzt in der Lage, das Verhalten von intelligenten Agenten mit komplexen Verhaltensmustern abzubilden. Zum einen ist die

Konvertierung von natürlich-sprachlichen Verhaltensbeschreibungen in rein numerische Systeme umständlich, da die Beschreibungen in qualitativer Form vorhanden sind. Zum anderen ist die Erstellung und Optimierung der mathematischen Modelle zeitaufwändig, und Fehler können bisweilen nur von Modellierungs-Experten gefunden werden. Ferner sind mathematische Modelle empfindlich gegenüber Datenausreißern und inadäquat wenn die Daten zu verrauscht oder unvollständig sind.

Um diesen Einschränkungen zu begegnen, habe ich ein auf Knowledge Representation und Reasoning basierendes Analyse-Framework entwickelt, um das Agentenverhalten mittels symbolischer Modelle von Bewegungen und Aktivitäten zu evaluieren. Der Schlüsselmechanismus innerhalb des Frameworks ist das Zuordnen von qualitativen Verhaltensbeschreibungen zu den abstrahierten Beobachtungen des Agentenverhaltens.

Basierend auf diesem Schlüsselmechanismus habe ich eine Reihe von Analyse-Methoden entwickelt, um das Verhalten der Agenten in der Domäne auf einer hohen Abstraktionsebene zu beschreiben. Damit begegne ich den Einschränkungen von rein numerischen Methoden der Modellierung.

Die vorgestellten Methoden werden in einer detaillierten Fallstudie aus der maritimen Navigation evaluiert. Dabei wird die Formalisierung der in der Seefahrt vorhandenen Verhaltensbeschreibungen untersucht und gezeigt, dass die resultierenden Modellbeschreibungen funktionsfähig sind. Zusätzlich wird gezeigt, dass das Framework mit unvollständigen, realen Daten umgehen kann und auch mit großen Datenmengen gut skaliert.

Contents

List of Figures

List of Tables

List of Algorithms

List of Diagrams

List of Abbreviations

AIS	Automatic Identification System
ASP	Answer Set Programming
COLREGS	International Regulations for Preventing Collisions at Sea
CTL	Computational Tree Logic
DCC	Double Cross Calculus
DRA	Dipole Relation Algebra
IMO	International Maritime Organisation
KR	Knowledge Representation and Reasoning
LTL	Linear Temporal Logic
MMSI	Maritime Mobile Service Identity
NMEA	National Marine Electronics Association
OPRA	Orientated Point Algebra
QR	Qualitative Reasoning
QRPC	Qualitative Rectilinear Projection Calculus
QSR	Qualitative Spatial Reasoning
QSTR	Qualitative Spatial and Temporal Reasoning
QTC	Qualitative Trajectory Calculus
RCC	Region Connection Calculus
SAT	Boolean Satisfiability Problem
UTM	Universal Transverse Mercator

Chapter 1

Introduction

Understanding complex behaviours is an essential component of everyday life, integrated into daily routines as well as specialised research. To handle the increasing amount of data available from (logistic) dynamic scenarios, analysis of the behaviour of agents in a given environment is becoming more automated and thus requires reliable new analytical methods. This thesis seeks to improve analysis of observed data in dynamic scenarios by developing a new model for transforming sparse behavioural observations into realistic explanations of agent behaviours, then testing that model in a real-world maritime navigation scenario.

1.1 Complex Behaviours in Operations Research and Logistics: Background and Motivation

Logistics and operations specialists are concerned with optimising the activities of agents (e.g., to reduce travel time, to minimise costs) and improving safety (e.g., to avoid risk of collision). A prerequisite for these tasks is the analysis of the observed domain. Such domains include maritime navigation, air traffic control, warehouse operations management, and cargo transport. Consider a maritime navigation example in which two ships are moving toward each other. Maritime logistics regulates ship movements to avoid collision, i.e., safety, while minimising course corrections, i.e., cost optimisation.

Two main elements are central to data analysis with the goal of optimisation and safety: agents[1] that execute movements or more complex activities, i.e., *agent behaviours*;

[1] agent: "anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators" (Russell and Norvig, 2003), page 32, whether humans or artificial agents.

and collected data about these behaviours, i.e., sensor data or *observations* about agents. Agents in this context include all goal-oriented acting entities, such as vehicles in traffic, pedestrians, etc. Each domain has its own semantic rule set that regulates interactions between agents, such as right-of-way rules or descriptions of polite behaviour.

Currently, mathematical methods are commonly used to analyse scenarios in operations research, while qualitative models are rare. These mathematical approaches gather numerical values observed from the domain and develop mathematical models, i.e., numerical formulas, differential equations, statistical models, etc., fitting the observed data.

However, conventional numerical tools for model generation are not well suited to the integration of mathematical models with semantic knowledge, which is a prerequisite for analysis. For reasons of efficiency, mathematical methods tend to avoid complicated models using too many variables. However, integrating more intelligent agents into a model forces the use of more variables. For example, in the maritime domain, rules are represented as the maritime rule set "International Regulations for Preventing Collisions at Sea", maintained by the International Maritime Organization (IMO) (2003) (IMO), and each vessel in the model must abide by them. Thus, generated numerical models tend not to take into account the complex semantic interaction rules between agents, which are inherent in real-world scenarios in logistics and operations research. This thesis develops a novel method for integration of readily accessible semantic behavioural information in a domain with observational data.

1.2 Example: A Collision-Avoidance Scenario

Consider an example of the maritime domain rules: two ships, with a heading toward each other, on a possible collision course, Figure 1.1. Ship A has ship B on its starboard side. Maritime domain rules state that ship A must manoeuvre to avoid the second ship B. The actors in this scenario are interested in a reduction in travel costs, i.e., not making unnecessary trajectory changes, as well as the safety aspect of avoiding collision. The safety aspect often occurs in combination with cost optimisation and relies more strongly on the situational understanding provided by knowledge of the domain rules.

In real-world scenarios in maritime environments, ships often reach their destination port without generating massive additional costs or colliding with one another. That is due to two factors:

1. Ship-controlling agents can sense the presence of other ships. This can be done via

optical means like human vision, or through technology supported by radar systems or the Automatic Identification System (AIS[2]) introduced by the IMO. Sensory means, however, bear the risk of errors.

2. To clarify actions to be taken, a rule set is necessary that covers situations of (potential) interaction. By integrating knowledge contained in that rule set with common-sense knowledge of physical laws, ship-controlling agents can estimate the actions and behaviours of other agents.

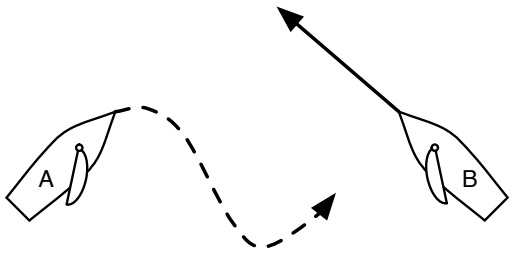


Figure 1.1 – Ship A on an avoidance trajectory and ship B keeping its course.

Agents controlling the ships assume that other ships' agents will avoid random movements and follow known domain rules. By integrating this knowledge with observational data on the other agents, they are able to generate the most likely hypotheses about the future behaviour of others and plan accordingly to reduce excess movements. Additionally, this gives them the opportunity to identify agents breaking the behavioural rules of the domain and react accordingly. For this reason, Pietrzykowski and Uriasz (2009) as well as Uriasz (2008) state that navigational knowledge is to be understood as a set of data, facts, rules, procedures, strategies, and theories combined with the interpretation and reasoning capabilities of the navigator, further strengthening the claim for the necessity of semantic integration in operations and logistics models.

1.3 Challenges in Operations Research and Logistics with Intelligent Agents

There are three essential challenges in the analysis of the behaviour of intelligent agents in operations research and logistics, particularly when applying a limited numerical modelling approach:

[2] `http://www.imo.org`, last verified October 19, 2015

1) Observational data gathered by various sensor devices are often noisy and incomplete for real world domains.

For example, when gathering observational data in the maritime scenario above, ships can be occluded by fog for visual confirmation or AIS messages might be overlaid by another radio signal, suppressing them. Thus, the means to handle incomplete or noisy observations must be integrated into its representation.

> **Research question I**
> *How can information available in the domain itself be represented for automatic qualitative analysis of dynamic scenarios, robust against noisy and sparse sensor data?*

2) Large amounts of data impede human operators in maintaining an overview of a scenario.

When confronted with a large amount of raw numerical data, such as generated by more than 50 other ships simultaneously in a maritime scenario, human operators struggle to maintain a complete overview of the situation[3]. Not taking the correct action leads to potentially dangerous collision scenarios, and suboptimal behaviour with respect to cost. The results presented by an analysis tool should have semantic meaning without need of further processing and must be able to highlight questionable behaviour that does not fit the domain rule descriptions.

> **Research question II**
> *How can analysis of dynamic scenarios be automated, such that the result contains high-level explanations of processes that occur around an agent?*

3) Mathematical models are based on metric observations and lack support for straight-forward integration of domain rules.

Mathematical models for operational research typically rely solely on observational data and disregard the semantics of domain rules, for example, when analysing ship pathways for fuel efficiency. It is generally possible to mathematically model semantic relations, for example with integer linear programming, but it is not easily understandable for humans and becomes complex to evaluate with increasing numbers of modelled features. Although modelling of right-of-way rules is possible for numerical models utilising mixed linear programming, scenarios featuring this method do not include large numbers of agents (Richards and How, 2002), are not feasible within a short time

[3]Information acquired interviewing Podebry (2014).

(Alejo et al., 2013; Omer and Farges, 2013), and for non-mathematical experts they lack intuitive readily understandable explanations for the found solutions.

That is, numerical models are not the most intuitive approach for modelling intelligent agent activities and behaviours from a human perspective, and therefore only provide limited traceability of explanations for optimisation and safety analysis. The ability to explain a found solution is often indispensable for understanding the problem as a first step toward a better system.

Behavioural descriptions for a scenario are typically provided either in rule form, for example, by the International Regulations for Preventing Collisions at Sea (COLREGS[4]) as a univocal rule set in the maritime domain, or as descriptions from domain experts (found in warehouse research (Berg and Zijm, 1999; Rouwenhorst et al., 2000)), both in natural language. Natural language descriptions might be vague and ambiguous, i.e., they are informal descriptions, precluding rigorous mathematical analysis.

The COLREGS are easily understandable for humans, but encoding them into mathematical formulas requires careful expert work and is not easily comprehensible for non-mathematical experts. Having a flexible representation for behavioural rules in domains eases the transfer between domains, in contrast to mathematical modelling, which requires careful crafting toward a single case. Also, it is difficult to convert natural language descriptions of agent behaviours into a system of purely numerical equations, as these so-called expert descriptions tend to be qualitative rather than numerical. Furthermore, generating these mathematical models is a time-consuming, tedious, and error-prone process.

Research question III
What formal modelling language is needed to represent behaviour in the context of a dynamic scenario more intuitively than that using pure numerical models, and how can observations be combined with rule descriptions to enable high-level semantic domain analysis?

I tackle these research questions within this thesis by introducing the following framework for the naval domain.

[4] `http://www.imo.org`, last verified October 19, 2015

1.4 A Knowledge-Based Framework for High-Level Agent Behaviour Analysis

To address the limitations of numerical methods, I developed a Knowledge Representation and Reasoning (KR)-based framework for analysing agent behaviour data utilising on symbolic qualitative models of the domain, i.e., agent movements and activities.

The discipline of Knowledge Representation and Reasoning is part of Artificial Intelligence research, covering the representation of information such that computer systems can reason about it in a formal way. Motivated by human cognition and human problem-solving strategies, KR focuses on planning and problem solving using qualitative, rather than quantitative, information.

Therefore, I argue for the introduction of so-called *transfer primitives* which form various spatial-temporal behaviours, like distinct movements (*forward, turn*). These ease the transfer of natural language descriptions of behaviours into formal descriptions. The goal of these transfer primitives is to convert natural language descriptions like "the ship moves forward and then executes a left turn" into a formal representation. In this case, transfer primitives denote a *forward movement* and a *turn left*, connected by a temporal transfer primitive like *next*. The key mechanism within my framework is an algorithm for *matching sensor data* of agents with the KR-based *descriptions of agent behaviours*.

My framework automates the process that a logistics expert would take in analysing a dynamic real-world scenario. In constructing the framework for automated high-level analysis of agents' action data, I combine descriptions of agent behaviour with formalised observations, compare Figure 1.2. Dynamic scenes are abstracted into a symbolic representation, then analysed based on known process specifications under the assumption of common-sense physics knowledge.

The combined descriptions and observations are instantiated by a logical reasoning process, resulting in high-level descriptions of the occurrences forming the central part of the analysis. Based on this key mechanism for matching, I have developed a range of analytical methods for investigating agent behaviour in a high-level way, which supports optimisation and safety analysis of operations research tasks involving intelligent agents. Thus, my framework addresses the limitations of a purely numerical modelling approach.

1.5 Case Study in Maritime Navigation

The framework introduced above for qualitative analysis is then evaluated in an extensive case study using actual maritime-navigation data. The case study is divided into three

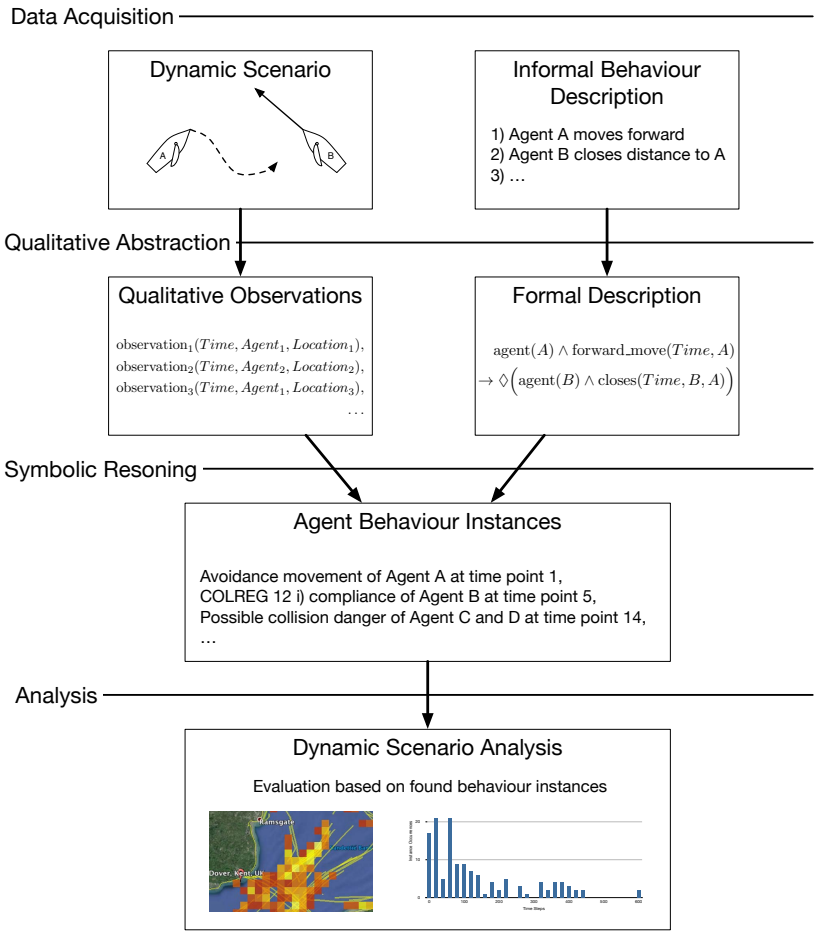


Figure 1.2 – Knowledge-Based Framework for High-Level Agent Behaviour Analysis.

aspects:

1. **Domain coverage and validity of formalised behaviour descriptions**
   To verify that the formalised behaviour descriptions are sufficient to analyse maritime scenarios, synthetically generated scenarios covering typical situations in the maritime domain are utilised. By correctly classifying the presented situations, the formalised rule coverage is demonstrated.

2. **Analysis of real-world sensor recordings in combination with formalised maritime rules**
   Real-world AIS data is analysed with the novel framework. Sample cases are highlighted and overall statistics shown.

3. **Scaling performance for large data sets**
   To ensure the applicability of the novel framework on large data sets, over 900 simultaneous ship trajectories are analysed over three days. To demonstrate its scalability, different numbers of ships are analysed and the computational time and theoretical space consumption on a test system displayed.

1.6 Contribution

This thesis gives one possible answer to the question of how a qualitative framework can be modelled for dynamic scenario analysis. Essential features include an intuitive method for the transfer of domain rules into a formal representation, while maintaining analytical possibilities for the domain based on spatial-temporal observations.

The main contributions of this thesis are:

- Providing a framework to transfer informal behaviour descriptions into formal behaviour descriptions, and sensory observations into symbolic observations. By analysing this qualitative information, a semantic model of an observed scenario can be generated. Building blocks of the framework are:
  - Transfer of metric observations into qualitatively represented observations.
  - Development of transfer primitives to bridge the gap from informal behaviour descriptions to formal behaviour descriptions for active agents in the scenario.
  - Methods to infer all instances of behaviours in a scenario based on descriptions and observations.
- A detailed case study analysing dynamic real-world maritime scenarios, based on formalised behaviour rules and abstracted observations.
  - Abstraction methods to transfer metric Automatic Identification System (AIS) messages into symbolic observations of ships.
  - Translation of the International Regulations for Preventing Collisions at Sea (COLREGS) into formal descriptions utilising the transfer primitives.
  - Implementation of a prototypical analysis system in Answer Set Programming.
  - Evaluation of the implemented framework in the maritime domain.
- Discussion of the application of hypothetical observations inferred from the available formal behaviour descriptions and observations in the case of incomplete sensor data.

- Discussion of the applicability of the introduced framework to other domains, based on the flexibility of the qualitative modelling approach.

1.7 Structure of this Thesis

The thesis is split into 6 chapters as follows:

1. **This Introduction.**

2. **Qualitative Temporal and Spatial Representation and Reasoning**
   Prominent methods from the field of qualitative reasoning are introduced, with a focus on spatial and temporal modelling as the basis for the use of transfer predicates.

3. **Abstract Observation and Formal Description Generation**
   In this chapter, transfer predicates are developed to simplify the translation of domain rules. The matching methodology, based on the transferred descriptions and the abstracted observations, is explained.

4. **Transfer of Maritime Domain Rules into Formal Descriptions**
   The chapter presents detailed descriptions of the transfer of domain rules into predicates for use in automatic analysis. It also includes a discussion of the use of deductively generated hypotheses to complete domain observations with sparse observational data.

5. **Evaluation in the Maritime Domain**
   The resulting novel representational and analytical framework is evaluated with a detailed case study in the domain of maritime navigation. It tests whether the framework is capable of handling real-world noisy sensor data, can scale to real-world amounts of sensor data, and is sufficiently expressive to formalise natural-language maritime collision-avoidance regulations.

6. **Conclusion and Future Work**

   The findings of this thesis are summarised and the outcomes compared with the proposed research questions from the first chapter. Results of the case study are evaluated, and the thesis closes with an outlook on further studies in the domain of qualitative analysis as well as ideas for further enhancements of this work.

Chapter 2

Qualitative Temporal and Spatial Representation and Reasoning

This chapter presents an overview of methods and techniques that can be combined to qualitatively analyse a dynamic scenario, as described in Chapter 1. To that end, formal representation and relation methods are introduced as the means for formalisation of the COLREGS (International Maritime Organization (IMO), 2003) and observations from the domain and as the basis for reasoning over the represented information. Formalisation of COLREGS is meant to translate the official regulations, denoted in natural language, to a verified formal framework on which the automated analytical system can be based (Kreutzmann, Wolter, Dylla and Lee, 2013).

In order to describe domain-dependent behaviours on a qualitative level, I first review advances in the field of qualitative representations, temporal and spatial, beginning with intervals and semi-intervals for time representation. From there, topologic, ordinal and orientation calculi are introduced. The representation section closes with an outlook on methods to describe qualitative change.

The next challenge is to reason over change, for example movements occurring over time. While integrating change into the qualitative representation itself is possible, techniques from the field of logic can provide more intuitive modelling of behaviours for domain experts. I further compare methods from the field of process recognition for their utility in analysis of dynamic domains. This chapter closes with an overview of process-recognition techniques and a description of model-checking as an alternative means to analyse qualitatively described dynamic scenarios.

2.1 Qualitative Representations

A qualitative representation of any real-world arrangement is an abstraction of the original information. The abstraction process aims to preserve the structure of the information to facilitate reasoning about it after the information has been reduced. Ideally, an abstraction process preserves only the pertinent information for the problem solution and omits anything unnecessary (Kuipers and Byun, 1991; Rajagopalan and Kuipers, 1994). This is closely related to the functionality of the human mind, as research on cognitive maps indicates (Kuipers, 2000). Humans are able to understand a sufficient part of the mechanics of many spatial/temporal phenomena to allow them to solve everyday tasks, such as kinetic energy in the interaction of physical objects or projection of movements (if the start conditions are known). For instance, when catching a ball that is thrown by moving the hand, it seems to be sufficient to know on an abstract level about the mechanics and parameters involved; details like the exact weight or the velocity vectors are abstracted when catching the ball.

The term **representation** is defined as a formal system to make explicit certain entities or types of information, together with a specification of how the system does this (Marr and Nishihara, 1978). The result of using such a representation to describe a given entity is a **description**. I make use of that fundamental principle by applying known qualitative representations and descriptions in chapter 3.

2.2 Representations, Reasoning, and Calculi

To clarify the terminology used in this thesis, I differentiate between the terms representation, reasoning, and calculi. A *qualitative representation* is an explication of relational, spatial or temporal knowledge between entities. Qualitative representations compare rather than measure. They aim to abstract away from any information not relevant to the task at hand. An example of a representation is a statement like "*event A happened before event B*" for time or "*point C is in front of point D*" for orientated points.

Qualitative reasoning, on the other hand, is a form of symbolic reasoning in which symbols act as semantic connectors between represented information. As above, the symbolic methodology aims at an abstraction of non-essential information. In general, qualitative reasoning describes a much broader field of reasoning techniques with specialised calculi. The subfield of *qualitative spatial-temporal reasoning* (QSTR) has been active since the 1990s and aims to develop lean and intuitive representations, which are usually aligned with human cognition. An example of qualitative reasoning is the connection of time

inference *(A before B)* and *(B before C)*, resulting in *(A before C)*. In plain language: given the knowledge that event A happened before event B and B happened before event C, one can infer that A must have happened before C.

Qualitative Calculi, as used in this thesis, are an existing formalism of QSTR. They are a specialised variant of qualitative reasoning used to explore previously defined operations for the given calculi. Usual operations for calculi are the generation of composition and converse relations, in other words, combinations and inversions of stated facts. An example of composition can be taken from the region connection calculus RCC8: *(A ntpp B)* $\circ$ *(B ntpp C)* = *(A ntpp C)*, meaning that a region A is within a region B and region B is within region C without touching borders. From this, it can be derived that region A is also in region C without connection points. An example of converse relations is: given *A ntpp B* we can state *A ntppi B*; by knowing that A is within B and not touching its borders, we can state that B is around A and not touching its borders.

2.3 Qualitative Models for Time Representation

Allen's Interval Algebra (Allen, 1983) and Freksa's semi-intervals (Freksa, 1992a) are noteworthy approaches to reasoning over time.

Allen's contribution consists of a temporal logic based on intervals and their qualitative relations over time. In contrast to a time-point-based approach that handles time as discrete and instantaneous observations, Allen's intervals allow for a more expressive description of a real-world situation than point-based descriptions, as most events related to time are not spontaneous.

By using 6 + 6 + ID relations between time intervals (*before, meets, overlaps, during, starts, finishes, equals*) and their 6 inverse relations (except for the equal relation, which would result in an identical relation; *after, met-by, overlapped-by, contains, started-by, finished-by*), 13 disjoint relations are introduced, compare Figure 2.1.

An example is the execution of an event. Allen's intervals allow for description of a scenario in which an event A started and ended before an event B, while event C was active the whole time and ended at the same time as B. In formal notation, that would read as in Figure 2.2.

Allen has shown that sound and complete reasoning within his descriptions is possible, although his algorithm as described in Allen (1983) was not able to find all inconsistencies within polynomial time. He provided composition tables containing all possible relation pairs for the interaction of 3 intervals,useful for lookup of results for time queries. A composition table forms the basis for qualitative orientation-based reasoning and is

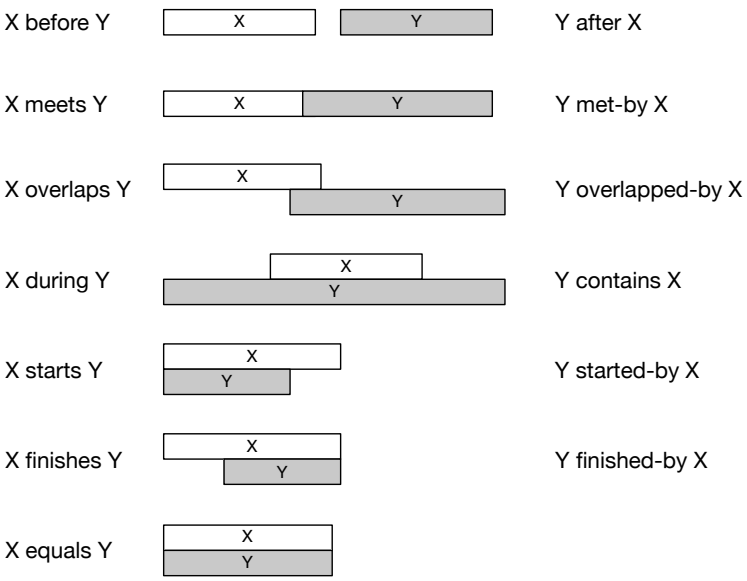
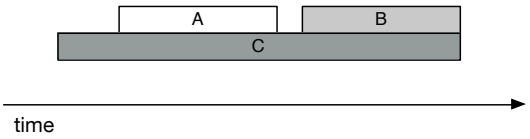


Figure 2.1 – Allen's temporal intervals (Allen, 1983).



$$A\ before\ B$$
$$A\ during\ C \qquad (2.1)$$
$$B\ finishes\ C$$

Figure 2.2 – Allen's temporal intervals example case, figure and notation.

arranged such that the rows and columns represent actual conceptual neighbourhoods. As Freksa (1992b) mentioned, a 2-dimensional table does not represent the structure of all conceptual neighbourhoods. A conceptional neighbourhood is given if there exists a direct transition from one qualitative relation into another.

When examined in detail, the proof for a subset of problems[1] might generate an exponential number of graphs to check, bringing the problem into NPSPACE[2].

Freksa (1992a) extended Allen's interval-based reasoning approach with the idea of conceptional neighbourhoods, enabling reasoning for scenarios with incomplete knowl-

[1] consistency checking for Allen representation graphs with multiple constraints can be converted into multiple equivalent graphs with single constraints

[2] The class PSPACE is the set of languages (strings associated with a problem) that can be decided with no more than a polynomial amount of storage space during the execution of the algorithm (LaValle, 2006), while the NPSPACE set consist of the languages that can be decided with a nondeterministic Turing machine that magically makes the correct choice.

edge, e.g. open intervals. Relations between pairs of semi-intervals result in conceptional neighbourhoods[3] if "they can be directly transformed into one another by continuously deforming (i.e. shortening, lengthening, moving) the events (in a topological sense)" (Freksa, 1992a). His rationale was that, for instance, it is enough to know that Newton's death took place before Einstein's birth to reason that Newton lived before Einstein. In this example, it is unnecessary to specify the start point of the interval of Newton's life or the endpoint of Einstein's. By the introduction of semi-intervals (open to one side) and their combination with the possible translations of relations in the conceptual neighbourhood, Freksa brought computational reasoning closer to humanlike natural incomplete-knowledge-based reasoning.

An approach for generalisation of the intervals into two dimensions was later done by Gottfried (2004), entering the spatial domain.

2.4 Qualitative Models for Spatial Representation

Qualitative spatial representations can be categorised as **orientational, topological, distance** or **panoramic** information. They provide the means to hold different spatial information depending on the model used.

A prominent example of an **orientation representation** is the approach of Freksa (1992b). Through the use of directional vectors on an orientation grid, it is possible to express relative relations in one of six regions or on one of the three lines themselves, resulting in 15 different qualitative relations. See Figure 2.3 and detailed description in section 2.4.3.

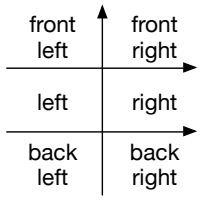


Figure 2.3 – Freksa's relative orientations (Freksa, 1992b).

A **topological representation** is part of the *Region Connection Calculus* (RCC) family, described by Randell et al. (1992). It is capable of describing connectivity and overlap of regions in different granularities. For the RCC8 calculus, 8 relations are defined: *disconnected* (DC), *externally connected* (EC), *partial overlapping* (PO), *tangential proper*

[3]Conceptional neighbourhoods can also be generated for other domains.

part (TPP) and its inverse (TPPi), *equal* (EQ), and *non-tangential proper part* (NTPP) and its inverse (NTPPi). Similar to Allen's interval composition table, Randell created these categories by combining two relations for three regions. Additionally, a neighbourhood-based transition graph, representing all possible transitions from one region connection description to the next, is well known in the QSR community (Sridhar et al., 2011) and shown in Figure 2.4.

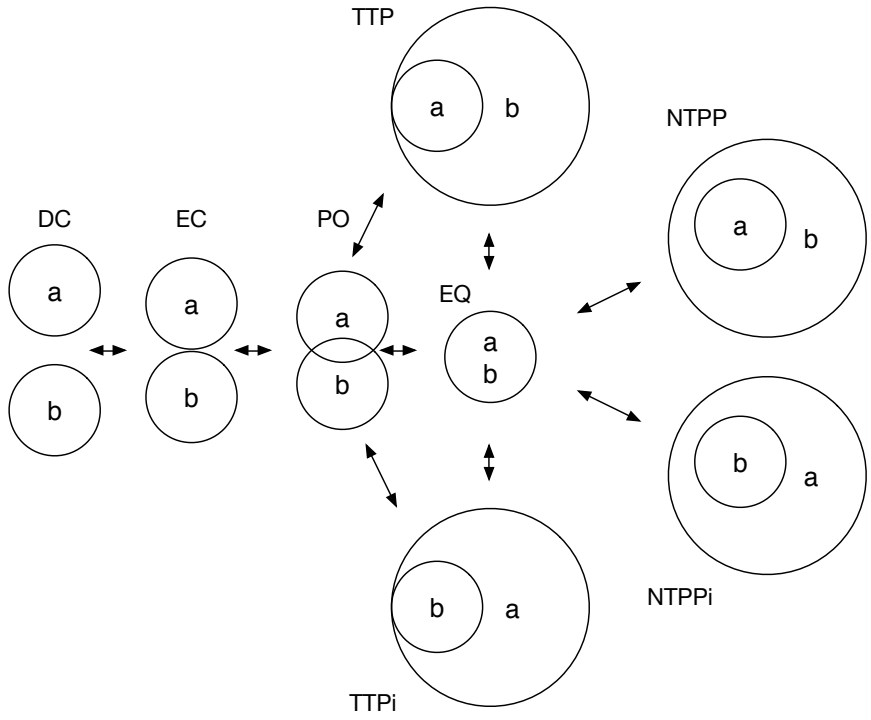


Figure 2.4 – The Region Connection Calculus (RCC-8).

Panoramic representation has been introduced by Schlieder (1991, 1993) as a panoramic view of point-based landmarks, representing an ordered arrangement of elements from a separated viewpoint. All reference elements are lined up on a horizon and thereby ordered. The amount of information can be extended by acquiring the ordering from the opposite position with respect to the elements. This representation has been extended by Wagner (2006) with metric information and qualitative directions to enhance robustness. Building upon that, Colonius (2009) added qualitative information about the relative perceived distance between landmarks, enhancing performance.

The work of Clementini et al. (1997) introduced a framework to describe 2-dimensional position information. In contrast to the previously introduced representations, orientation

as well as coarse **distance information** is represented. Therefore they use relations such as *front, right, back, left and far or close* as depicted in Figure 2.5.

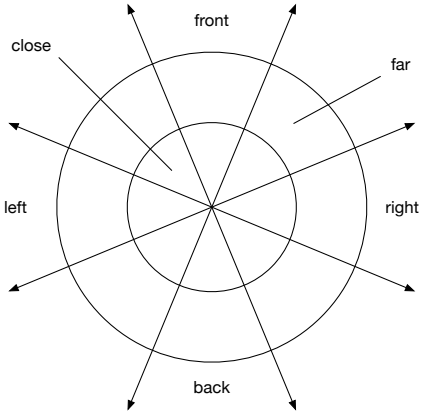


Figure 2.5 – 2-dimensional categories as in Clementini et al. (1997)

In the following section, the orientational calculi most relevant to the development of the Orientated Point Algebra $\mathcal{OPRA}_m$ calculi are described, based on the assumption that moving agents have an egocentric view of the world. Such a view can be represented intuitively as orientated point-based information and is thus nicely suited for the challenge of representing agent behaviour in the present task. We start with relative orientation calculi that encode the view from an observer relative to an object and increase the granularity. For a more complete survey, see Chen et al. (2015). The representations introduced here are either binary relations between three points of interest or ternary relations. Binary relations can be utilised to represent the relative location of two entities such as points, lines, or intervals without referring to a third entity; Allen's intervals are binary. Ternary relations, however, can directly capture the relevant frame of reference which occurs in natural language semantics (Dylla, 2008; Chen et al., 2015; Wallgrün et al., 2007). By describing an arrangement of points from a third point, statements like $left\ of$ are possible. These are well suited to describe the rules for behaviour.

2.4.1 Flip-Flop Calculus

The Flip-Flop calculus was introduced by Ligozat (1993) and makes use of a directed line between two points as the reference system for a third point.

Possible relations for the third point are: $left$ of the line and $right$ of the line, or on different segments of the line as directly in $front$, directly $behind$ or $in\ between$. Additionally, there are two further relations on the $start$ and end points of the line itself:

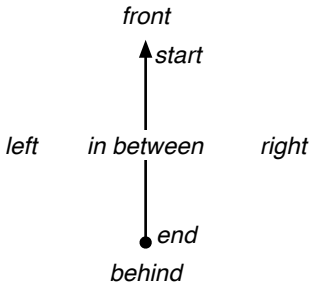


Figure 2.6 – The $Flip - Flop$ calculus

one if the third point coincidences with both of them and one if not.

2.4.2 Dipole Relation Algebra (DRA)

The Dipole Relation Algebra was introduced by Moratz et al. (2000). This full relational algebra method for inference is based on the use of dipoles in the neighbourhood-based approach of Schlieder (1995). By construction, this relation is binary, as a reference is given between two lines A and B.

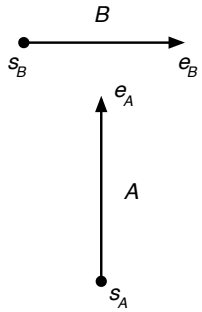


Figure 2.7 – The $Dipole$ Relation Algebra

The Dipole Relation Algebra adapts the $left, right$ notation of Ligozat by comparing the $start$ and end points of two line segments, allowing the user to describe relations between named directed lines in the form of four $left, right$ descriptions. As an example, in Figure 2.7 the relation is *A lrrr B*, which is described as: line A is defined by the starting point s_A and the endpoint e_A and analogous line B is defined by s_B and e_B. Point s_B is left of the line A, point e_B is right of line A, point s_A is right of line B and point e_A is also right of line B.

2.4.3 Double Cross Calculus (DCC)

The ternary representation of the Double Cross Calculus is based on cognitive principles and aims for an abstraction of orientation similar to the previously introduced approaches. In comparison with the $\mathcal{LR}$ calculus and the *Single Cross Calculus* introduced in (Freksa, 1992b), the additional regions allow for a finer description. The Double Cross Calculus is created by combining two Single Cross representations and result in descriptions named $front$ (f), $right-front$ (rf), $right-neutral$ (re), $right-back$ (rs), $back$ (b), $left-back$ (ls), $left-neutral$ (le), and $left-front$ (lf), as well as the regions in between. Freksa (1992b) reviews several approaches to qualitative spatial reasoning and creates an inference scheme similar to Allen's (Allen, 1983) temporal interval logic.

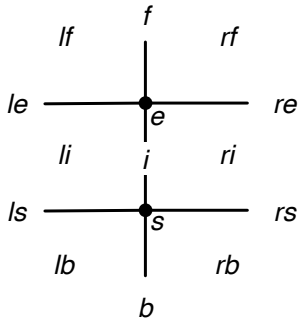


Figure 2.8 – The Double Cross Calculus

The Double Cross representation is based on three points, which induce a partitioning of space, see Figure 2.8.

2.4.4 Orientated Point Algebra ($\mathcal{OPRA}_m$)

In contrast to the rather coarse granularity of the previously discussed representations, the binary $\mathcal{OPRA}_m$ representation as introduced in Mossakowski and Moratz (2012) is a point-based orientated algebra with a scalable granularity m. Mossakowski and Moratz developed tables[4] and the proof of correctness for the algebra in their paper, and recommended it as especially useful for navigational problems.

$\mathcal{OPRA}_m$ utilises half-lines and angular sectors to make a distinction between positions seen from one orientated point to another. The granularity is given by the number of half-lines ($2m$) and angular vectors ($2m$), in other words, an evenly sectioned space

[4] A (precompiled) composition table consists of all consistent atomic formulas for an algebra and can be used for decisions of correctness of a given scenario, for example.

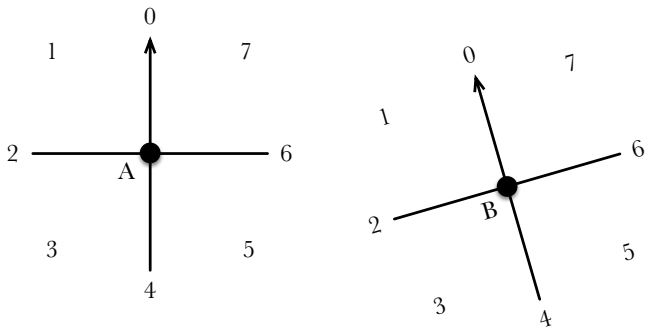


Figure 2.9 – $\mathcal{OPRA}_m$ representation with m = 2.

formed by m lines that cross the point of interest, as seen in Figure 2.9 and 2.10. A common $\mathcal{OPRA}_m$ notation for a configuration of two orientated points, X and Y, is written as $X\ {}_m\angle^l_k\ Y$, where k denotes the sector in which X is placed as seen from Y, l the sector in which Y is seen from X and m denotes the granularity of the spatial intersections that form the sectors. In both figures, the physical arrangement of the two points is identical; the relation for m = 2 is $A\ {}_m\angle^1_5\ B$ and for a granularity of m = 4 is $A\ {}_m\angle^3_{11}\ B$ (same relation).

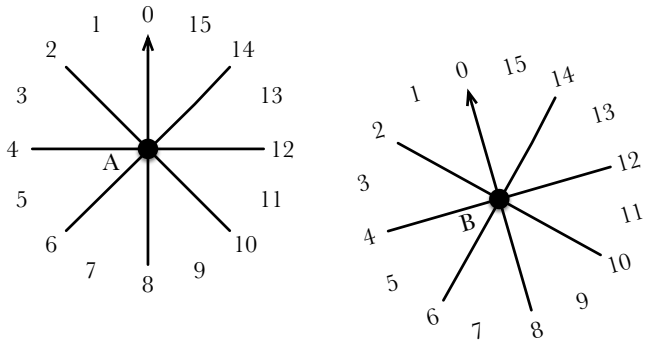


Figure 2.10 – $\mathcal{OPRA}_m$ representation with m = 4.

The latter relation has an accuracy, for the angular sections, twice as fine as the former, due to having twice the number of sections. On the downside, this makes computation more complex in terms of time and space, as more cases must be examined. One other noteworthy case covered by $\mathcal{OPRA}_m$ relations occurs when both relevant points have an identical position: the notation is then $A\ {}_m\angle i\ B$, where i is the direction of B as seen from A.

2.5 Qualitative Models for Representation of Change

In this section, two qualitative calculi are introduced that inherently model spatial and/or temporal change.

2.5.1 Qualitative Trajectory Calculus (QTC)

The Qualitative Trajectory Calculus (Van de Weghe, 2004; Van de Weghe, Kuijpers, Bogaert and Maeyer, 2005) describes motion as the difference between two positions of an object at different time points. This is done by comparing the configuration of two objects k, l at a current time point t and the time points t^- immediately before that, which may then be used to predict the distance between the two objects immediately after the current time point, namely t^+.

When the representation is based on the $\mathcal{DCC}$, Van de Weghe calls it QTC$_C$ and differentiates between object movements in the qualitative values of moving away, toward each other and stable (+,-,0). Whereas the basic QTC only considers the change in distance of an object pair, the extension based on $\mathcal{DCC}$ also considers direction, albeit consuming more computational power in doing so. Orientated on directional calculus with the possibilities to move along each axis, Van de Weghe creates 81 possible relations (3 possible distance changes on 4 axes = 3^4). For the full description see (Van de Weghe, Cohn and Maeyer, 2005).

Based on this qualitative description, it is possible to encode a car-overtaking scenario, for example, as sequences of QTC$_C$ relations. The overtaking example would look like a concatenation of relative distance transformations in chronological order (with $\rightsquigarrow$ as the symbol denoting a following state):

$$\{(-+00) \rightsquigarrow (-+-+) \rightsquigarrow (00-+) \rightsquigarrow (+--+) \rightsquigarrow (+-00)\} \tag{2.2}$$

This example has a change relation for each relevant change in the arrangement of the two cars. After a study of the method, it is possible to apply it to a given spatial temporal scenario, although it takes considerable abstraction for the inexperienced eye and is promoted by the authors for use in expert systems.

2.5.2 Qualitative Rectilinear Projection Calculus ($QRPC$)

The Qualitative Rectilinear Projection Calculus of Glez-Cabrera et al. (2013) features a similar approach to the QTC method. Again, the dichotomy of $left$ and $right$ and $front$

and $back$ is used to express relational knowledge about two entities. A distinctive feature of the QRPC is the abstraction from trajectories. The authors generalise a vector from the current heading of a point traveling a (curved) trajectory. From here, 4 elements of the notation are derived:

1. (P_iP_j) is the relative disposition between the oriented rectilinear projection of the object i and the oriented rectilinear projection of object j.

2. $(O_iO_j^{LR})$ is the relative position of the first object with respect to the $left-right$ dichotomy of both objects.

3. $(CO_i^{FB})(CO_j^{FB})$ denotes the relative position of the intersection point C of both trajectory-based vectors of the points.

4. $(O_iO_j^{FB})$ is the relative position of object i with respect of the $front-back$ relation of object j when the trajectories are superimposed.

By use of this notation and the derived conceptional neighbourhood table, the authors are able to describe complex motion patterns involving two objects between two given states as a sequence of QRTC relations. Theoretically, an interesting application of the system lies in the prediction of spatial temporal patterns, as all possible outcomes of a partial action are included in the conceptional neighbourhood graph. Although the authors note that the description can be verbalised in natural language, the abstraction applied here is on a similarly complex level as the QTC approach, making it tedious to use for a domain expert from another discipline.

2.5.3 Summary of introduced qualitative representations

This section started with qualitative representations of relative-position calculi. After introducing the underlying concepts of directional classification, it followed their evolution until the introduction of the multi-granular $\mathcal{OPRA}_m$ calculus. $\mathcal{OPRA}_m$ offers sufficient expressibility for the challenge addressed in this thesis because a flexible granularity allows for a representation tailored to the domain. Further complex calculi were introduced that can cope with change of time and spatial relations. In spite of their inherent ability to express change, their application to real-world data is no longer intuitive and might require the attention of a modelling expert, which contradicts the aim of this thesis, to create a simple straight-forward transfer of natural language domain rules into formal expressions.

Because of the complex models necessary for representations including temporal constraints, this thesis argues for the decoupling of the temporal aspect from the representation layer into the agent behaviour description. Furthermore, the work of Dylla (2008) expresses the usability of the $\mathcal{OPRA}_m$ calculus, more explicitly a combination of $\mathcal{OPRA}_m$ and the specialised Alignment Calculus $\mathcal{LA}$, to handle moving points in an agent control framework. It is noteworthy that the work of Dylla differs significantly from this thesis, as it is aimed to provide a representation suitable for agent control, in contrast to the current focus on the analysis of a dynamic environment with an strong aspect of generality. Furthermore, (Kreutzmann, Wolter, Dylla and Lee, 2013) hints at the applicability of a spatial representation in combination with logic to handle naval domain rules.

2.6 Logics

Having established the methods to represent spatial information, I now address the logical means of describing the relations and changes encoded in that information. Logic can be used as a mathematical expression to state knowledge about relations between facts. Originally based on the Greek word logos, translated as "the reason", the use of logic in information science is normally formal. The following section explains the basic notation and functions necessary for this project.

2.6.1 General Logics

In its simplest form, a logic statement is a proposition that is either *true* ($\top$) or *false* ($\bot$)[5]. Two-valued logical systems are composed of atomic facts, a syntax that defines how facts can be combined into more complex statements like the logical conjunction ($\wedge$) or the disjunction ($\vee$), and semantics comprising a set of rules that define how the formulas evaluate to *true* or *false*. If a formula evaluates to true, the system is referred to as a model. If there is no model for a given set, a formula is unsatisfiable.

This project requires the expression of knowledge about space and time, and I have chosen to utilise logic to that end. To ensure a consistent understandable syntax, I first explain crucial concepts of logic, then demonstrate how Linear Temporal Logic will be sufficient to achieve the project objectives.

Propositional Logic contains the most commonly known elements of logical reasoning. The set of propositions Φ contains all $\phi_1, \phi_2, \phi_3, ..., \phi_n \in \Phi$ with $n \in \mathbb{N}$; either $\phi_n = \top$

[5] or, as a special case, it can be not set, if there is insufficient knowledge to make a statement.

or $p_\phi = \bot$ is set, but never both at the same time. It uses the binary connectivities and ($\wedge$), or ($\vee$), and $implies$ ($\rightarrow$), as well as the unary not ($\neg$) for negation. Propositional logic is able to evaluate a combination of connected statements, depending on the validity of the propositions used. An example logic statement is: fact a is $true$ and fact b is $false$; if they are combined with the $\wedge$ operator and the $\neg$ for b, the result is also $true$, formally written as $a \wedge \neg b = \top$.

In a **monotonic logic**, derived facts hold true, even in the case that additional clauses are added. In other words, if a formal logic is monotonic, adding a formula to the pool of formulas does not reduce the set of derivable facts. Monotonic logics are unable to reason by default, meaning that a fact in a monotonic logic can either be proven true or false. In a **non-monotonic logic**, derived facts can reduce the set of consequences of the formulas. Examples of non-monotonic logics are default reasoning, abductive reasoning, and belief revision systems.

Default reasoning (Reiter, 1980), as the name implies, allows the solution of a formula to result in true by default and is non-monotonic. This leads to a generalization, which can result in a reduction of true clauses when additional formulas are added. For example, "Boats typically float" is a default statement that can be seen as a formula. Any test of floating for a boat entity will return true. By adding the clause "sunk boats do not float", the set of consequences is reduced.

In contrast to a logic in which rules either apply or not, in **defeasible logic** (Nute, 1989) a priority ordering of rules is given. Strict rules specify that a fact will always follow as a consequence of another. Defeasible rules indicate that a fact is a typical consequence of another, similar to default logic rules, and undercutting defeaters specify exceptions to defeasible rules and can inhibit their application. As an example, Tachmazidis et al. (2012) introduces a combination of mapping and non-monotonic reasoning techniques to deal with large data sets[6].

An existential expansion of the expressibility of logic formulas is developed under **modal logic**. By adding base operators for the expression of a *necessity* ($\Box$) and a *possibility* ($\Diamond$)[7], one can create statements like "it is possible that a is true". This also results in the potential for modal temporal operators, such as "it was the case that a was true". A widely known statement in modal logics is,

$$\Diamond p \leftrightarrow \neg \Box \neg p$$

[6]They essentially use a divide-and-conquer approach that includes parallel reasoning over facts while using defeasible logic techniques to allow rules organised by a superiority relation.

[7]Please note that the operators are named to aid understanding and are not equivalent to the operators used in Linear Temporal Logic in Section 2.6.2.

which reads as "the proposition p is possible if and only if it is not necessary that p is not true." Along the same lines, the alternative can be stated as "the proposition is necessary if and only if it is not possible that p is not true", which can be written as

$$\Box p \leftrightarrow \neg \diamond \neg p$$

.

2.6.2 Temporal Logics

Extending propositional logic with the functionality to express time constraints, like *always, eventually* and *until*, leads to the foundation of **temporal logic**. By integration of ideas expressing a temporal component, statements such as $a = \top$ can still be a valid truth assignment, but that assignment can change over time. Temporal logics make use of operators from propositional logic and modal logics to express their relations.

Whereas temporal logic always has the ability to reason over time, **Linear Temporal Logic** (LTL) (Pnueli, 1977) restricts the reasoning to one time line and does not allow for branching over multiple time lines. Linear Temporal Logic adds four basic operators to the common logic syntax: $next$ ($\bigcirc$), $always$ ($\Box$), $eventually$ ($\Diamond$) and $until$ ($\phi \mathcal{U} \phi$). Please note the difference from the modal logic operators.

The $next$ operator requires a true value for ϕ at the next discrete time step, $always$ denotes the fulfilment of ϕ for the entire subsequent path, and $eventually$ denotes that ϕ holds somewhen on the subsequent path. $Until$ is considered an expression for a truth statement that holds for a time point until a given time point with a defined stop condition. LTL enjoys widespread use, for example, in robotics (Antoniotti and Mishra, 1995) controller specification by a correct-by-construction manner (Kress-Gazit and Wongpiromsarn, 2011), in motion planning (e.g. Kloetzer and Belta, 2006; Smith et al., 2010; Lahijanian et al., 2011), and in real-world robotic applications (Kloetzer and Belta, 2010).

Further, the widely known **Computation Tree Logic (CTL)** (Clarke et al., 1986), belonging to the family of branching time logics, permits reasoning over many executions at once. It is understandable as a tree-like structure that allows branching with elapsed time. A good way of describing the difference between LTL and CTL is that LTL checks all LTL properties of a trace for satisfaction (and does that for all traces), while CTL verifies the formulae in its tree structure. If one is to verify the liveliness of a system[8], CTL would

[8]Checking, for example, if a software program might reach states that cannot be left again, i.e., 'stall'.

be an intuitive approach, whereas to check for instantiated behaviour descriptions in an already recorded observation base, LTL can be applied more straight forward. Both logics belong to the family of CTL* which allows for a combination of both methodologies, resulting in complex description properties. CTL is an often-applied logic for system verification in various disciplines such as Model Checking in biology (Kim et al., 2008), Signal Timing Verification (Dasgupta et al., 2002). It is used to test system behaviour given an initial state, tracing all possible translations of states allowed by the logic relations.

2.6.3 Summary of Logics

For the task of modelling representations of agent behaviour that denote **change over time and space**, Linear Temporal Logic offers convenient modifiers. Similar to (Kreutzmann, Colonius, Wolter, Dylla, Frommberger and Freksa, 2013), the motivation to use LTL is twofold. First, LTL allows for adequate basic process recognition and understanding, while remaining close to existing models for robotic reasoning, entities that are grounded in a time- and space-defined environment of relevance for process recognition. Second, because of its similarity to natural language descriptions, LTL provides a basis for domain experts to describe agent behaviour without the need for in-depth informatics knowledge. The process of description can be eased further, as explained in Chapter 3. The rich semantics of CTL are not necessary for the given task, as neither planning on multiple paths nor branching is needed for the analysis task itself.

A simple behaviour description in terms of *'first condition one holds, then condition two holds*' is formally expressible and simultaneously understandable by someone who is not a logic expert. Because the **modal** property is inherent to LTL, the **monotonic** property allows the implementation of this method in dynamic environments. This is relevant for real-world applications that are in need of situational understanding. Having determined the means to represent relations in Section 2.5 and now the means to describe agent behaviours, all that remains is to find an efficient method to match the observed data to the provided descriptions.

2.7 Process Recognition

The multitude of diverse approaches to the problem of process recognition can be categorised into learning approaches, probabilistic process descriptions, and logic-based declarative approaches.

As summarized in Colonius (2012), examples of the learning approaches are Markov networks (Bennewitz et al., 2005; Liao et al., 2007), Bayesian networks (Yang, 2009), supervised learning (Balcan and Blum, 2010), and inductive logic programming (Dubba et al., 2011), all of which require a training phase before deployment. Evolutionary methods also exist for collision-free navigation (Smierzchalski and Michalewicz, 2000; Szlapczynski and Szlapczynska, 2012). Similar to Kreutzmann et al. (2011), this thesis prefers an approach that does not need a training phase. This enables the user to pose dynamic queries to the knowledge base in a flexible formal language and to integrate new knowledge at any point in the process. Possible solutions are found with declarative, logic-based formalisms (Forbus, 1996; Bredeweg and Struss, 2004). These are already in use, for example, for the integration of ontology-based knowledge to recognise contexts in an ubiquitous environment (Mastrogiovanni et al., 2009) or in controller construction for robot control.

Combinations of logic and uncertainty are also in use. Fagin et al. (1988) applied a metric basis for qualitative measurements, while Ilic-Stepic (2010) used qualitative probability to compare the likelihood of facts against each other and Goldszmidt and Pearl (1996) used qualitative uncertainty to model if/then clauses. For a comprehensive overview of the differences between descriptions of uncertainty in possibilities, probability theory, and multi-valued logics, see Dubois and Prade (2001).

In the process-recognition approach of Kreutzmann, Colonius, Wolter, Dylla, Frommberger and Freksa (2013), observations were matched against pre-proposed processes, which is essentially a model-checking approach. Model checking is common practice in software verification and robotics. Planning instances in this field can be found in Cimatti et al. (1997); Edelkamp and Jabbar (2006); Kloetzer and Belta (2010).

An exemplary use case for the application of symbolic model checking, including the use of Linear Temporal Logics (LTL) for the modelling, was done by Halle et al. (2013) to automatically check for anomalies in large firewall-protected computer networks. There, LTL is used iteratively and while no model of a valid action has been found, no anomalies have been observed. Once the model checker fails, an incoherency in the firewall rules has been found and is reported.

2.7.1 Implementations of Logic Model Checking

Generally speaking, the task of model checking is to establish whether, given a model M with a state s in M and a formula ϕ, s satisfies the formula ϕ. In the context of spatial-temporal logic, model checking is the task of searching for a sequence of actual

spatial-temporal transitions in s fitting a description of a rule formula ϕ, as in Kreutzmann, Wolter, Dylla and Lee (2013).

Logic languages are well suited for handling qualitatively represented information. One of the widely known logic languages is Prolog, developed in 1972 by Robert Kowalski, Alain Colmerauer, Philippe Roussel, and others. Prolog allows the user to state declarative clauses and handles tasks by proving that a given clause holds in a given domain. Despite the advantages of a simple input language and effective inference mechanisms, Prolog has the downside of being able to enter infinitive loops if the implementation permits it to do so, which is non-trivial to predict.

There are several other declarative Prolog languages, like Datalog, SWI-Prolog or BProlog, but for this thesis, Answer Set Programming (ASP) was chosen because of its efficiency and maintained tool suite, Potassco, the Potsdam Answer Set Solving Collection (Gebser et al., 2011).

An Answer Set Program can be viewed as a set of statements describing objects of a domain and their relations. This is very suitable for the given task of identifying which descriptions of agent behaviour are matchable to given atomic parts of our so-called observational data, or in other words, an answer set. ASP syntax is close to Prolog, but the programming style differs significantly. For example, because ASP solves tasks as a conversion of search problems to computed stable models, it will terminate in cases in which Prolog may enter infinite loops.

An ASP program consists of statements in the form of rules with a head and a body. The head is a predicate that holds true if all conditions in the body evaluate to true. If no condition is given, the head automatically evaluates to true and is called a fact. Syntactic sugar is present, too, so one can describe multiple holding facts as so-called choice rules. For example, given the fact that $conditions$ 1, 2 and 3 hold $true$, we can infer by application of ASP's mechanisms, that $head(X)$ will also evaluate as true, because all elements of its body are fulfilled. Formally written:

$$\begin{aligned} &head(X) :- condition(1) \wedge condition(3). \\ &condition(1,2,3). \\ &X :- \top. \end{aligned} \tag{2.3}$$

Besides its simple syntax, ASP offers clear and transparent mathematical semantics. Its non-monotonic feature differentiates it from classical logic by allowing reduction in the fact base when new knowledge is added. This eases the task of knowledge representation (Gelfond and Kahl, 2014). Our task of model checking can be straightforwardly

implemented in ASP. Running an ASP program includes two phases, $grounding$ and $solving$. First the so-called $grounding$ takes place, in which all variables are grounded with all fitting facts. Current answer-set solvers work variable-free, which makes the preparation step necessary. The grounder chosen is for this project $gringo$. For $solving$ the answer set, $clasp$ is deployed, a solver that utilises conflict-driven no-good learning, a technique that has proven very successful for satisfiability checking (SAT), as written by the Potsdam group[9].

A more in-depth explanation is available in Lifschitz (2002, 2008).

2.8 Chapter Summary

This chapter presented an overview of representations for qualitative spatial and temporal relations. Starting with the temporal relations of Allen, the chapter described the evolution of orientated point-based calculi and gave an overview of more complex calculi integrating spatial and temporal components. For the analytical challenge of this thesis, the $\mathcal{OPRA}_m$ representation was chosen as it is able to encode spatial relations in sufficient detail while still maintaining an intuitive syntax.

Logical elements necessary for the methods deployed in this thesis were introduced, starting with basic First Order Logic then explaining the necessary properties to reach Linear Temporal Logics. Process recognition techniques were presented with a focus on the challenge of analysing a given domain. After naming relevant probabilistic and symbolic techniques, the latter were selected as conforming to the preconditions of qualitative modelling outlined in Chapter 1.

The chapter closed with a description of the mechanics of Answer Set Programming as the basis for the following chapters.

[9] `http://sourceforge.net/projects/potassco/files/potassco_guide/`, checked May 20, 2014.

Chapter 3

Qualitative Analysis Framework

This chapter formally introduces the agent behaviour analysis, describing the methods deployed and the elements that form the basis for the agent behaviour descriptions embedded in the analysis framework. Based on Linear Temporal Logic (LTL), a seamless integration between representation and reasoning is demonstrated for (1) identification of patterns to search for, (2) generation of hypothetical facts to fill observation gaps, and (3) analysis of the domain by matching descriptions to qualitative facts in the knowledge base. Introduced in chapter 2, LTL and $\mathcal{OPRA}_m$ provide the formalisation of the framework. LTL offers the means to describe temporal processes and $\mathcal{OPRA}_m$ is used for qualitative spatial representation. Spatiotemporal primitives for behaviour description are based on named qualitative representation and temporal logic. The chapter begins with an overview of the framework, continues with the definition of terminology and the formalisation of primitives (e.g., left of, in front, or movement), then describes the grounding of the observation data set from the domain. Finally, analysis and hypothetical knowledge generation are presented.

3.1 System Overview

The framework tackles the challenge of analysing a dynamic scenario based on informal descriptions of agent behaviours.

As the task requires observational information from the given domain, a suitable representation structure is necessary to hold the information. The representation enables integration with the behavioural descriptions for matching. A second requirement is a component for transfer of informal behaviour descriptions into formal behaviour rules

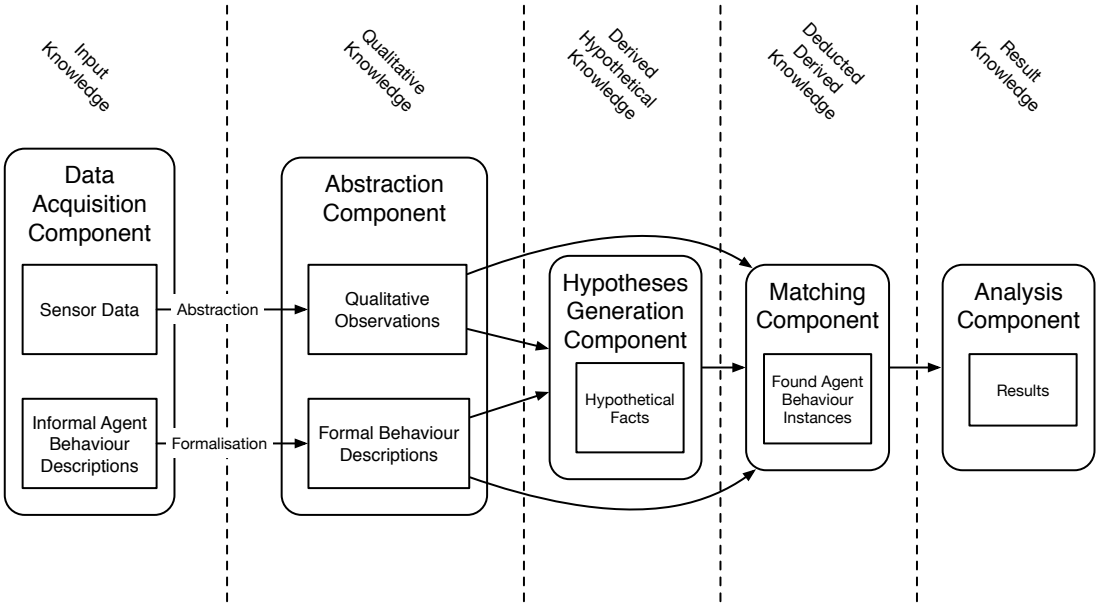


Figure 3.1 – Framework overview.

encoding possible agent actions. Sensory observations are abstracted into the same representation. Also, matched instances must be represented in a form suitable for later analysis. The framework consists of 5 components, as seen in Figure 3.1. Each component in the figure holds its related data and offers it to the next component for further computation. The following is an overview of the components, which are described in detail in the Sections 3.3 to 3.8.

1. **Data Acquisition Component**

   The purpose of this component is the recording of all essential information for the subsequent analysis. Fundamental building blocks for analysis of a dynamic domain are (1) information about agent actions in the observed domain and (2) descriptions of events that take place. The input information for this component is typically provided via sensors for the agent actions. Sensors commonly record data in metric values, for example the position and orientation of an agent in 2-dimensional coordinates and a vector. Input data in the form of descriptions of agent behaviours are typically provided by domain experts in narrative form or by publicly available domain rule sets in text form describing possible behaviours, as,

for example, the COLREGS in the naval domain.

As the procedure of data gathering is not in the focus of this work, it is described as a use case in Sections 4.1, 4.3 and 5.2.

2. **Abstraction Component**

   To set the measurements from the observations and the informal descriptions of behaviours in relation to each other, any of a multitude of different approaches, metric as well as qualitative, could be used (compare Section 2.7). I argue for a qualitative spatial-reasoning approach in combination with temporal logics, utilising intuitive descriptions and enabling the application of efficient matching algorithms. The qualification component ensures transfer from metric measurements into qualitative facts suitable for logical reasoning and transfer from informal narrative behaviour descriptions into formal rules.

   Therefore, two sub-components are required, the transfer from *measurements into facts* and from *narrative descriptions into formal rules*, both of which are a form of abstraction. The part that transfers observations from metric measurements to qualitative facts does so by mapping each measurement point (e.g., for locations) to a qualitative identifier, see Section 3.3. An example of this is the transfer from GPS position measurements into named location identifiers. If the information needed for the matching process is encoded in an abundance of measurements, clustering methods can be deployed on the metric values to reduce the number of qualitative facts (Kreutzmann et al., 2011). The previously introduced spatial (3.4.3) and temporal (3.4.3) primitives are used for the transfer of behaviour descriptions into formal rules (Section 3.4) that can be matched with the qualitative facts derived from observations.

3. **Hypothesis Generation Component**

   Once appropriate representations have been generated for observations as well as description rules, the framework can detect "irregularities", such as avoidance movements of agents without a clear explanation, if suitable descriptions of actions are provided. This can be extended, depending on the scenario, by generation of hypothetical facts, such as the hypothesis that another agent, undiscovered by the sensors in the domain, is present and caused the unexplained behaviour. The hypothesis-generation component takes as input the abstracted qualitative facts and

descriptions to generate hypothetical facts, which can be inserted into the matching component to support the matching process and provide additional information for the analysis component, see Figure 3.2. The full description of this component is given in Section 3.6.

4. **Matching Component**

   The matching component (Section 3.7) is the final prerequisite for the analysis component of the framework. By fitting abstracted observations into the formalised rules encoding the known agent behaviours for the domain, all instances of observed agent behaviours are returned and thus a high-level interpretation of the vast information set is achieved. For each instantiated behaviour description, start and end times are noted, as well as participating entities and eventual spatial points of interest. This forms the basis for the analysis through correlation of behaviours with respect to execution time and participating agents. Inputs for the matching component are the qualitative facts, optional hypothetical facts, the formalised agent behaviour descriptions and possibly common sense inferences (for example, the fact that ships are physically present even if no observation is recorded for a time point). The outputs of this component are called instantiated behaviour descriptions and provide data for high-level analysis.

5. **Analysis Component**

   The final component is dedicated to the analysis of the agent behaviour instances found by the matching component. Found instances can be processed for visual presentation, sorted and clustered for a more intuitive interpretation, and single instances subjected to an in-depth evaluation. Based on the inherent characteristics of the data, appropriate displays can be created, like a heat map visualising the spatial distribution of found instances or a temporal arrangement in a histogram.

   This is done to enable an analyst to comprehend the inner workings of the observed scenario more intuitively than a large data dump would allow. Even blind spots, spatial or temporal regions where no agent behaviour instances were found, can hold important information, such as indications that sensors are not functioning properly or relevant agent behaviour descriptions are lacking. A detailed description can be found in Section 3.8 below.

3.2 Information Flow in the Framework

A typical application of this framework is found in the naval domain. It is used here as an example case for information flow through the framework. The data acquisition component is provided with (1) **sensor observations** in the form of an information set for agents consisting of positional information, orientation with reference to a global system (reference frame), an identifier, and a time stamp, and (2) **behaviour descriptions** as a set of informal rules, regulating navigational behaviour when two ships come into relevant distance of each other, see Figure 3.2.

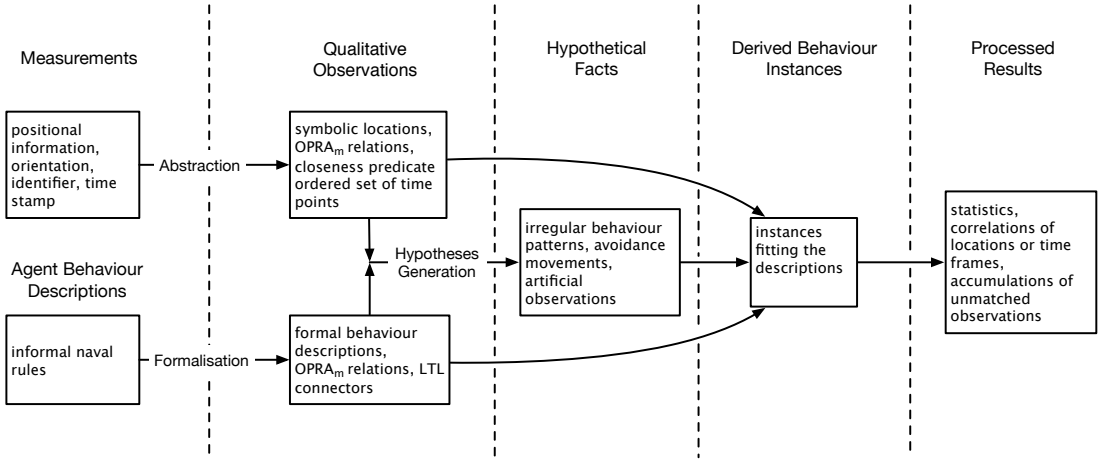


Figure 3.2 – Information flow overview.

Also visible in Figure 3.2 is the information flow in the other components. The abstraction component has two qualification tasks: (1) abstract sensor measurements, such that metric coordinates become **symbolic locations** (section 3.3), orientation between ships is represented as $\mathcal{OPRA}_m$ relations, a predicate describing proximity between two ships is generated if the distance is under a given threshold, and the time stamp is sorted into a ordered set of time points. Further, (2) the informal rule set is converted into a set of **formal behaviour descriptions** by decomposing the provided interaction rules between naval agents. This is achieved with the assistance of the description primitives, representing characteristic movements, and the qualitative spatial configurations connected by temporal logic, representing change over time.

Once abstracted observations and formalised behaviour descriptions are present, the hypothesis generation component can extract irregular behaviour patterns, such

as avoidance movements without a discernible cause, and utilise them for hypothesis generation, inserting hypothetical presences of unseen ships or hindrances into the observation fact base. A further method to generate hypothetical data is to allow artificial observations based on the last recorded (temporal or spatial) observations, inserting **hypothetical positions** for the time frames for which no information is available for the agents.

In the matching component, abstracted observations and any generated hypothetical information are compared with agent behaviour descriptions and all instances fitting the requirements, i.e., all combinations of observations that suffice to instantiate a formal description, are collected. This is the critical step in analysis of dynamic scenarios. Start and end times of **found instances** and participating agents is noted.

The analysis component summarises the instances of the behaviours found and aims to present the data in an intuitive way. Additions include **statistics regarding correlations of locations or time frames**, accumulations of unmatched observations, agents that are engaging others more frequently, or other relevant analytical elements.

3.3 Observation Abstraction Component

In the first part of the abstraction component from Figure 3.1, qualitative abstraction defines the translation from a spatio-temporal configuration, a.k.a., our scenario instance, into a qualitative description. For instance, the GPS position of an entity at a given time point can be translated into a qualitative observation *obs(location, time point)*, composed of a *location* and a *timepoint* at which the observation was made. A *location* in this context is a qualitative named identifier and a *timepoint* is the qualitative identifier denoting a point in the time frame of the observed scenario. Built on the approach of (Kreutzmann, Colonius, Wolter, Dylla, Frommberger and Freksa, 2013), this thesis represents time as an ordered sequence of independent worlds, or time points, in the logic model. By reviewing them in temporal order, the change, delta, between single worlds represents the change over time. For example, if a $shipA$ is in $location_1$ at $timepoint_1$ and in $location_2$ at $timepoint_2$, then the change is the movement from $location_1$ to $location_2$ if the locations are different. The delta between time points is the information that enters, in abstracted form, the observation base. It does so as atomic propositions which we call facts.

A transfer from metric measurements, in the example of positions, into abstract facts is defined as follows: for each position estimate (assuming a 2-dimensional plane) $pos(x, y) \in \mathbb{R}^2$ a discrete qualitative location $location_i \in \mathcal{LOC}$ is created for all time points i, with $i \in \mathbb{N}_0$. $\mathcal{LOC}$ stands for the unity of all locations. Formally:

$$pos(x,y)_i \rightarrow location_i \;\; i \in \mathbb{N}, location_i \in \mathcal{LOC} \tag{3.1}$$

Furthermore, two elements are relevant for the generation of abstracted facts. (1) The qualitative fact base consists of static facts, which means that, once entered, facts do not change over time (monotonic). Admitting new facts is possible, of course. Thus we build agent behaviour detection on a solid base without having to consider compromising derived facts later on. (2) Each time a transition of an abstracted fact is made, it contains a timestamp. To achieve this, admittance of facts is encapsulated into observations. An example observation is presented as a combination of a time point, an entity, and a location, heading, or both.

$$observation(timepoint, entity, location, heading) \tag{3.2}$$

A fact F is mapped to an instance I of an observation O from the observation scenario S. The set of all facts is named $\mathcal{F}$. An observation o is formally defined as a set of instances $\{I_1, ..I_n\} \in \mathcal{I}$ with $n \in \mathcal{N}$ that hold true for the fact F, $O(I_1..I_n) = \top$.

3.4 Behaviour Formalisation Component

In the second part of the abstraction component from Figure 3.1, the key component for the qualitative analysis is description of the processes that occur in the domain. To ease the combination of representation and reasoning, I deployed a logical symbolic method for description of the processes.

3.4.1 Formal Specification of Agent Behaviours

An agent behaviour description $\mathcal{ABD}$ is defined as a set of predicates $\mathcal{P} = \{P_1, \ldots, P_n\}$. A predicate P is composed of a number of m-ary domain tuples or grounded facts $P = (U_1 \times \ldots \times U_m, F_1, \ldots, F_n)$ with the domains $U_1, \ldots, U_m$, $F \in \mathcal{F}$ and $m, n \in \mathbb{N}$. The m-ary domain tuples U with the logical constraints form the relations between the variables for each process. An example of this is the predicate denoting that two ships are in danger of collision, in which the predicate is named $couldCollide(x_s, y_s)$, with $\forall x_s \in U_s$ and $\forall y_s \in U_s$ where U_s denotes the domain of all ships.

A set of facts $\mathcal{F}$ is composed of all abstracted observations from the given scenario. Each fact F is defined as $F(T, I)$ with $T \in \mathcal{T}$ as the set of all time points $\mathcal{T}$ on which an observation has been made and $I \in \mathcal{I}$ as the set of all instances in which an observation was made $\mathcal{I}$, as defined in section 3.3.

$$\mathcal{ABD} = (P_1, \ldots, P_n), P \in \mathcal{P} \tag{3.3}$$

and

$$\begin{aligned} & P = (U_1 \times \ldots \times U_m,), \text{with domains } U_1, \ldots, U_m \\ & \vee P = (F_1, \ldots, F_n), F \in \mathcal{F} \\ & \vee P = (U_1 \times \ldots \times U_m, F_1, \ldots, F_n) \end{aligned} \tag{3.4}$$

and

$$F = (T, I), T \in \mathcal{T}, I \in \mathcal{I} \tag{3.5}$$

The top-level agent behaviour descriptions represent the rules provided by the domain expert and are hierarchically decomposable until they are described on the level of observed facts from the knowledge base, compare Figure 3.3. The provided high-level behaviour descriptions can be found in the present data. The decomposition predicates are elements of the high-level process descriptions. The predicates themselves are composed of either a number of other predicates or facts, which are abstracted observations.

This reduces the task of deciding if a process has occurred based on the available data to a matching problem in the search tree of the hierarchically decomposed process description. For example, if it is to be decided if a movement has occurred for an agent in the domain, the decomposition splits the *movement* predicate into a *location change* predicate, a *time change* predicate and an *entity* predicate, representing the acting agent. This is further decomposed into the respective facts, as a *location change* and a *time change* require observations of that agent at two different locations at two different time points.

From here, the elements of the agent behaviour descriptions are split into two groups to ease definition: first time constraints and their integration, then spatial descriptions.

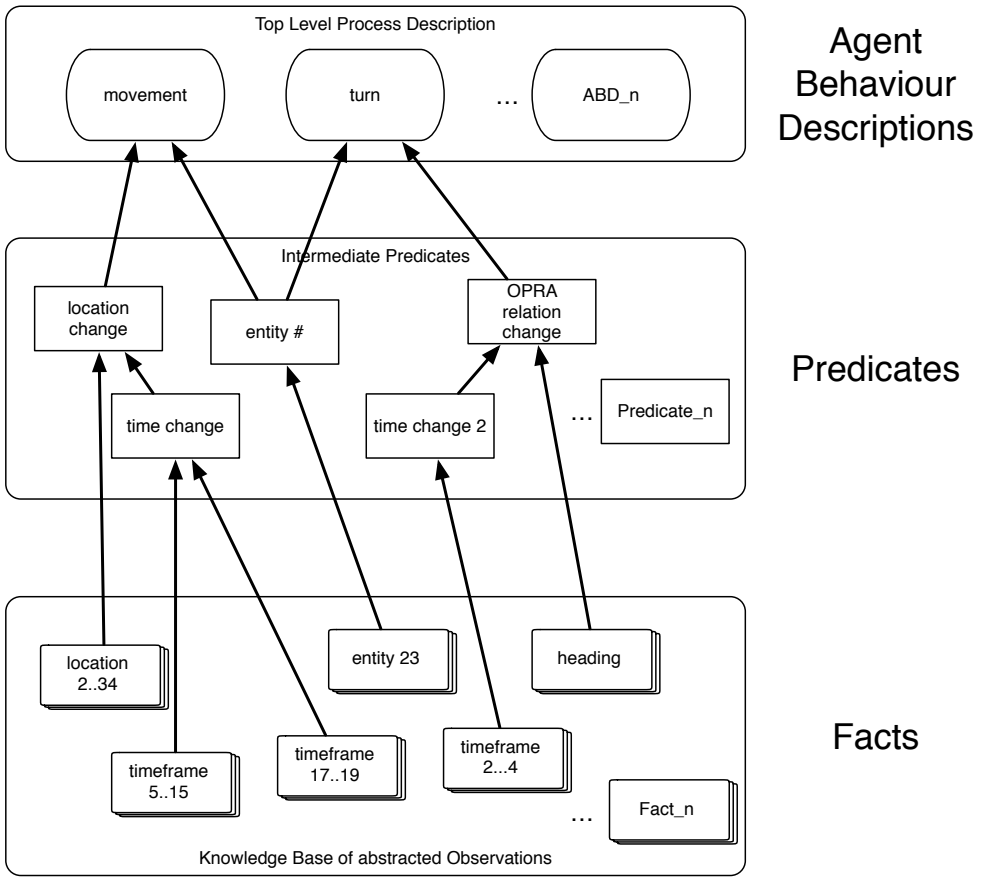


Figure 3.3 – Systematic overview of hierarchical description decomposition.

3.4.2 Temporal Descriptions

Time is formally modelled using Linear Temporal Logic to describe the truth of predicates over time. It is modelled as a linearly ordered path of states (i.e. no branching, non-cyclic).

While Linear Temporal Logic as presented in Section 2.6.2 already brings its own set of temporal operators, this thesis encapsulates them for easier use by non-experts. Therefore the following predicates are introduced, using time points $T, T' \in \mathcal{T}$ and ϕ as the symbol for a predicate P that can $hold$ if the conditions are met, thus evaluating to $true$. In detail, if $\forall condition_1, ..., condition_n = \top$, then $holds(condition_1, ..., condition_n) = \top$ with $condition_i \in \{\top, \bot\}$ and $i, n \in \mathbb{N}$.

It is to be noted that the introduced predicates are all based on discrete time intervals, similar to Allen and Hayes (1985). As the high-level predicates for the agent behaviour

descriptions denote a (timed) sequence of spatial arrangements, as seen in section 3.5, a time point marks a discrete point in the time line at which the spatial configuration holds. A next point in the timeline follows after an interval, only specified *next* based on the fact that no other observation has been made in between. Thus each predicate cannot range over a time interval but rather over a set of time points, each in turn connected to a world state in the form of observations related to that single time point. Time point $T \in \mathcal{T}$ is assumed to be the set of all time points $\mathcal{T}$ as used in Section 3.4.1.

$next(T, T', \phi)$ holds true, if and only if the condition ϕ is fulfilled[1] in the next available T' time point, i.e., world state. Note that the time points are linearly ordered, so the subsequent time point is always the time point $+1$.

$$\begin{aligned} next(T, T', \phi) := \quad & holds(\phi, T) \\ & \wedge holds(\phi, T') \\ & \wedge T' = T + 1 \end{aligned} \tag{3.6}$$

$after(T, T', \phi)$ holds true, if and only if the condition ϕ is fulfilled in a time point T' that is temporally ordered behind the current time point T.

$$\begin{aligned} after(T, T'), \phi) := & holds(\phi, T) \\ & \wedge holds(\phi, T') \\ & \wedge T < T' \end{aligned} \tag{3.7}$$

$before(T, T', \phi)$ holds true, if and only if the condition ϕ is fulfilled in a time point T, i.e., world state temporally ordered before the current time point T', or, as is equivalent, the opposite of $after(T, T')\phi$, so $before(T, T')\phi := after(T', T)\phi$.

$$\begin{aligned} before(T, T', \phi) := & holds(\phi, T) \\ & \wedge holds(\phi, T') \\ & \wedge T < T' \end{aligned} \tag{3.8}$$

[1] Note that the $next$ ($\circ$) operator from LTL is not the same as the $next(T, T', \phi)$ operator introduced here.

$always(T, \phi)$ holds true, if and only if the condition ϕ is fulfilled in the current time point T, i.e., world state and all temporally ordered later time points.

$$\begin{aligned} always(T, \phi) \coloneqq \quad & holds(\phi, T) \\ & \wedge holds(\phi, T') \\ & \wedge \forall T' > T \end{aligned} \tag{3.9}$$

$until(T, T', \phi)$ holds true, if and only if the condition ϕ is fulfilled in all time points T', i.e., world states that came temporally ordered before the current time point T and remained fulfilled.

$$\begin{aligned} until(T, \phi) \coloneqq \quad & holds(\phi, T) \\ & \wedge holds(\phi, T') \\ & \wedge \forall T' < T \end{aligned} \tag{3.10}$$

$between(T, T', \phi)$ holds true, if and only if the condition ϕ is fulfilled in all time points known between and including T' and the current time point T, with T' before T.

$$\begin{aligned} between(T, T', \phi) \coloneqq & \; holds(\phi, T) \\ & \wedge holds(\phi, T') \\ & \wedge holds(\phi, T'') \\ & \wedge T' < T \\ & \wedge T'' < T' \\ & \wedge T'' > T \end{aligned} \tag{3.11}$$

3.4.3 Spatial Descriptions

As spatial configurations are already an integral part of the environmental representation, introduced in Section 2.4.4, the following section defines basic predicates for use in the analysis. I first denote two directional primitives, $ahead$ and $back$. Following their definition, the base predicates $left$ and $right$ are set. Later in Section 3.5, a definition for a $movement$ and a $turn$ make use of the previously defined temporal predicates

and the spatial descriptions. More complex predicates follow with the definition of a $forward\ move$ and a $turn\ left$ and their respective alternatives. These predicates form the basis for agent behaviour descriptions and aim to significantly ease the transfer from domain rules to descriptions for domain experts. Relations are defined over the spatial domain of points in 2D Euclidian space $\mathbb{R}^2$. In the following, the symbol $orientatedPoint$ is utilised to denote the heading and position of an orientated point in the 2D plane and the symbol $location$ to define an abstracted position on the plane.

Both the $ahead$ and $back$ primitives are based on two $\mathcal{OPRA}_m$ orientated points, however only the orientation of the first is used.

$$\begin{aligned} &ahead(orientatedPoint_1, orientatedPoint_2) := \\ &\qquad orientatedPoint_1 \ {}_m\angle_x^{0..m/2} \ orientatedPoint_2 \\ &\qquad \vee orientatedPoint_1 \ {}_m\angle_x^{7*m/2..2*m} \ orientatedPoint_2 \end{aligned} \tag{3.12}$$

As depicted in Figure 3.4, for a given granularity m, an ahead predicate is defined as one orientated point positioned in a 45-degree cone ($^1/_8$ of a circle) in either derivation of the frontal side of the other point. Important here is that for an $\mathcal{OPRA}_m$ relation, two points are necessary. The figure shows one of the two points in the centre of the intersecting lines for the segments and the other placed anywhere in the indicated (red) cone.

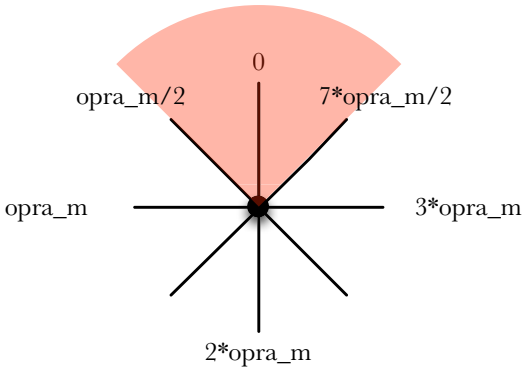


Figure 3.4 – This figure demonstrates the $ahead$ predicate. All points located within the red cone are ahead with respect to the centre of the sectioning lines.

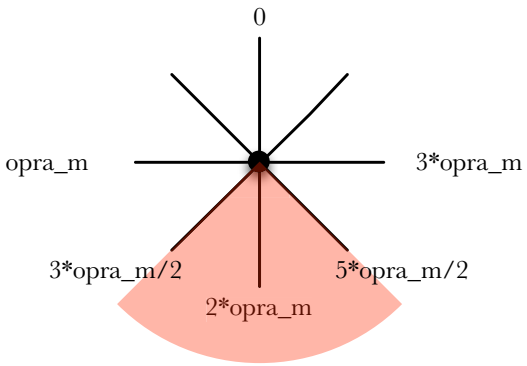


Figure 3.5 – This figure demonstrates the $back$ predicate. All points located within the red cone are in the back with respect to the centre of the sectioning lines.

Here, the orientation of the point in the cone can be ignored, as it functions only as

the relation for the other. For denoting the changeable granularity in the figure, $opra\_m$ is four (number of sectioning lines) but can be larger if a finer sectioning is desired. For the purpose of the introduced predicates, four suffices[2].

The $back$ primitive is defined similarly, with the difference that the cone is orientated to the back of the focal point, as seen in Figure 3.5.

$$back(orientatedPoint_1, orientatedPoint_2) := \\ orientatedPoint_1 \; {}_m\angle_x^{3*m/2_{..5*m/2}} \; orientatedPoint_2 \tag{3.13}$$

Analogous to $ahead$ and $back$, $left$ and $right$ predicates are defined, compare Figure 3.6 and 3.7

$$left(orientatedPoint_1, orientatedPoint_2) := \\ orientatedPoint_1 \; {}_m\angle_x^{0..2*m-1} \; orientatedPoint_2 \tag{3.14}$$

$$right(orientatedPoint_1, orientatedPoint_2) := \\ orientatedPoint_1 \; {}_m\angle_x^{2*m..4*m-1} \; orientatedPoint_2$$

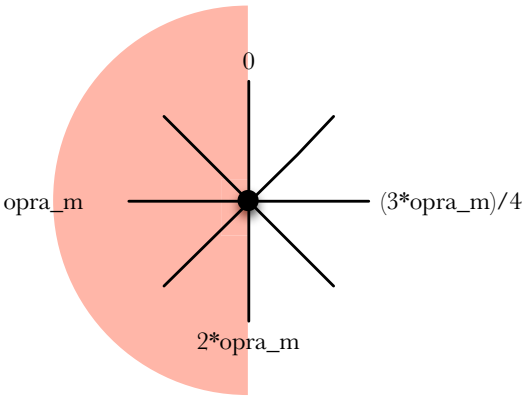


Figure 3.6 – $\mathcal{OPRA}_m$-conforming representation of the left side of a given point, depending on granularity level.

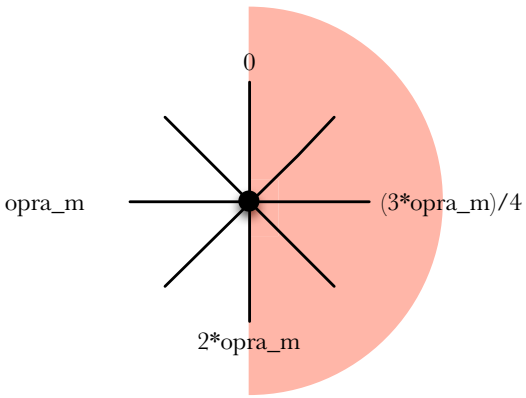


Figure 3.7 – $\mathcal{OPRA}_m$-conforming representation of the right side of a given point, depending on granularity level.

[2]Please note that the granularity of the $\mathcal{OPRA}_m$ predicate may be finer than $\frac{m}{2}$, but not coarser to avoid loss of expressiveness.

3.5 Change Descriptions

Two basic predicates denoting change, $movement$ and $turn$, can now be defined. The $movement$ predicate is a general element to denote a change in location. As our observations are grounded in real-world domains, it is reasonable to suppose that entities take time to relocate. Thus, a movement requires two different locations at different time points.

$$\begin{aligned} movement(location_1, location_2, T, T') \coloneqq \\ location_1 \neq location_2 \\ \wedge\, after(T, T'). \end{aligned} \tag{3.15}$$

The $turn$ predicate denotes a change in orientation. It can happen simultaneously with a movement, but it is not required to do so. Similar to Equation 3.15, the change is denoted by two different orientations and it is assumed that a turn takes time to execute. Here we make use of the $orientation$ symbol, which represents only the orientation of a given point. It can be substituted by $orientatedPoint$ if the positional information of the point is disregarded.

$$\begin{aligned} turn(orientation_1, orientation_2, T, T') \coloneqq \\ orientation_1 \neq orientation_2 \\ \wedge\, after(T, T'). \end{aligned} \tag{3.16}$$

Derived from these elementary predicates, a further set of simple predicates is defined to help human operators translate their domain knowledge into spatial temporal formalisms. A directed movement is defined through the inclusion of $\mathcal{OPRA}_m$ elements, allowing for a description with flexible granularity. For example, the $forward\ movement$ is defined by a direct move in the current heading. A forward motion does not have to be restricted to a movement exactly on the 0-line; depending on the scenario, a cone can be used in relation to the granularity. The only indispensable requirement here is that the user must avoid overlapping definitions for forward movements and other movements.

$$
\begin{aligned}
forward\ move(location_1, location_2, orientatedPoint_1, orientatedPoint_2, T, T') :=& \\
& movement(location_1, location_2, T, T') \\
& \wedge ahead(orientatedPoint_1, orientatedPoint_2) \\
& \wedge after(T, T').
\end{aligned}
\tag{3.17}
$$

Here $m > 2, m \in \mathbb{N}$ is a granularity for an $\mathcal{OPRA}_m$ predicate.

Similar movements, like $backwards$, $sidewards$ or $diagonal$, can also be described with $\mathcal{OPRA}_m$ predicates. For example, backwards would require the $\mathcal{OPRA}_m$ predicate $location_1\ {}_m\angle^0_{m*2}\ location_2$.

Directed turns are describable, as well. Again, the $\mathcal{OPRA}_m$ predicate is used with the range of m as a definition for $left$ with $m_l = \{1, .., (2 * m)\}$.

$$
\begin{aligned}
turn\ left(orientatedPoint_1, orientatedPoint_2, T, T') :=& \\
& left(orientatedPoint_1, orientatedPoint_2) \\
& \wedge after(T, T').
\end{aligned}
\tag{3.18}
$$

Turns to the right are analogically represented by $orientation_1\ {}_m\angle^1_{2*m}\ orientation_2$.

The defined predicates allow for a high level of coarseness, as the specified time intervals are open to one side, similar to Freksa (1992a).

3.6 Hypothesis Generation Component

As the system is intended to work on real-world data, it is sensible to assume gaps in the observation database as well as misinterpretations of phenomena that occur. The method of qualitative abstracted descriptions applied in this thesis is to some degree robust against gaps in the observations because only one observation for each critical part of the agent behaviour description is necessary for matching. In the case that such robustness is not sufficient, hypothetical facts may be introduced. Two methods for developing them are within the scope of this thesis. (1) A proximity-based approach to generate hypothetical facts which are inserted into the knowledge base with a reduced certainty about the fact, as introduced in Colonius (2012) for the warehouse logistic domain. This approach requires a modified reasoning to take into account the integration

of certainties for the qualitative facts and is prototypically implemented in that paper. (2) If the domain is not well suited for the proximity-based integration of hypothetical facts, an alternative method is abductive knowledge generation based on provided agent behaviour descriptions for abnormal behaviour. Abnormal behaviour for an object in a domain must be specified formally, similar to an agent behaviour description, with the difference being its ability to generate new facts to reason about. In contrast to the insertion of likelihood-rated facts to ease the analysis process, abductive generated facts are dealt with as true or false, greatly easing integration into the reasoning process.

3.6.1 Abductive Generation

Given an agent behaviour description ABD and an incomplete set of observation facts $\mathcal{F}$, hypotheses of observation facts $F \notin \mathcal{F}$ which are not in the observation base but are needed to explain ABD can be inferred to close gaps in the analysis of the system. Abduction is thus used to find an explanation for F_i. New facts are derived from a known consequence to fit the description. This task depends on suitable agent behaviour descriptions $ABDH$ to direct the hypothesis generation.

Here, I differentiate between abductive logic reasoning, as described above and applied in (Santos and Shanahan, 2002; Bhatt and Dylla, 2009), and explanation generation, which attempts to understand inference processes and their traversal through the reasoning path (Frausto et al., 2008) or is interleaved with generalisation to elucidate fitting descriptions of agent behaviours (Hirsh, 1987).

The requirements are:

- a set of agent behaviour descriptions for hypothetical fact generation $\mathcal{ABD}_H = ABD_1, ABD_2, .., ABD_n$
- a set of facts $\mathcal{F} = F_1, F_2, .., F_n$.

From these, a set of hypothetical facts $\mathcal{H}$ is defined which includes all newly derived facts. The new facts conform to the definitions of the observational facts. Thus, they are indistinguishable from the sensor-abstracted facts from the point of view of the reasoning process. This requires a careful modelling of the agent behaviour descriptions used for the fact generation, as incorrectly inferred facts falsify the analysis.

3.6.2 Proximity-based Generation

Another way to create qualitative hypothetical data is proximity-based generation. As described in Colonius (2012), a new data point is constructed based on relative closeness

or proximity of qualitative facts, either spatial, temporal, or a combination of both. In contrast to the abductive generation of facts, which requires an agent behaviour description of abnormal behaviour, no behaviour description is needed for the proximity-based approach. The requirements for proximity-based generation are:

- a set of spatially and/or temporally related facts $\mathcal{F}_{ST} = F_1, F_2, .., F_n$
- a domain-dependent predicate expressing the $proximity$ of facts.

Defining a set of hypothetical facts $\mathcal{H}$ is similar to the abductive approach. A vital difference is that the quality of the generated facts cannot depend on the validity of an agent behaviour description on which the fact is based. Insertions of proximity-based generated facts reduce the validity of the analysis because the method may insert necessary but previously missing facts as well as surplus of facts which are purely hypothetical but never actually occurred. The methods for proximity-based generation cope with that feature by propagation of a qualitative certainty measurement which is also integrated into the analytical reasoning. The following formula C maps an atomic part, for example a fact ϕ of a logic agent behaviour description, to a uncertainty value (decreasing certainty):

$$C(\phi) = \{true, certain, uncertain, \\ doubtful, false\} \tag{3.19}$$

In Colonius (2012), the combination of two atomic facts ϕ, ψ results in the minimum certainty of the single facts for the initial step. This approach is chosen based on the principle of simplicity.

$$C(\phi \wedge \psi) = min(C(\phi), C(\psi)) \tag{3.20}$$

While that is possible, it creates the implication that the end result of a qualitative analysis is no longer either true or false but may be something in between. On the upside, the introduction of a qualitative expression for certainty of facts allows a discernible measurement of inserted facts. For example, spatial closeness can be handled for an object o and locations $l_i, l_j, l_k, l_m \in \mathcal{LOC}$ when observations of named objects at location $l_i \in \mathcal{LOC}$ are present and a closeness predicate is defined:

$$C(at(o,l_i))\begin{cases} true & \text{if } at(o,l_i) \\ certain & \text{if } close(l_i,l_k) \wedge at(o,l_k) \\ uncertain & \text{if } close(l_i,l_k) \wedge close(l_k,l_m) \wedge at(o,l_m) \\ doubtful & \text{if } \neg at(o,l_j) \\ false & \text{if } at(o,l_j) \wedge l_i \neq l_j \end{cases} \tag{3.21}$$

Here, observation $at()$ of a fact close to l_k is still rated $certain$ and an iteration one more close() relation away (l_m) is rated $uncertain$. If there is no observation of an entity at a given time point at any location in the set of locations $\mathcal{LOC}$, a default hypothesis is generated for all locations with a rating of $doubtful$. A $false$ rating is given if the observation has been made at the time step somewhere else and is usually not explicitly stated in an implementation, as negation is defaulted if not otherwise specified.

Temporal predicates are dealt with similarly. A working integration of the rated hypothetical facts into the matching process is shown in Colonius (2012).

3.7 Matching Component

The matching component takes as input the qualitative facts, either abstracted from sensory observations or generated by the hypothesis component and the formalised agent behaviour descriptions. As the behaviour descriptions are constructed of spatial and temporal configurations that refer to agents, their interactions and actions, as well as environmental features, they can be instantiated if all parts are present as observational instances.

So, the challenge of recognising agent behaviour descriptions in a vast data set is characterised by finding the set of agent behaviour descriptions ABD that includes all predicates P that evaluate to true when combined with the set I of observed facts of the domains. I solve this by deploying model-checking techniques.

Take as an example the situation of two sailing ships A and B from the introduction, Figure3.8. A possible agent behaviour description *change_course*, Equation 3.22 (omitting the necessary spatial relation for simplicity) is decomposed into predicates describing the relative *wind* direction.

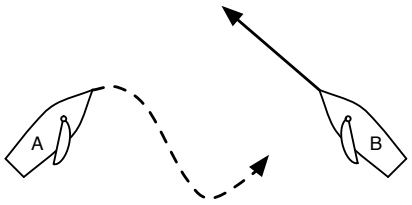


Figure 3.8 – Ship A on an avoidance trajectory and a second ship B keeping its course.

$$\begin{aligned} &change\_course(vessel(A)) := \\ &\qquad ships\_approaching(vessel(A), vessel(B) \\ &\qquad \wedge risk\_of\_collision(vessel(A), vessel(B)) \\ &\qquad \wedge wind\_direction(vessel(A), portside) \end{aligned} \tag{3.22}$$

There is a *risk of collision* based on a qualitative predicate indicating a distance threshold and a *ships approaching* predicate indicating that at least one of the ships moves toward the other. This allows a hierarchical decomposition of the agent behaviour as depicted in Figure 3.9. Similar to that decomposition, unchanging agent behaviours such as *keep_course* are then creditable without much effort.

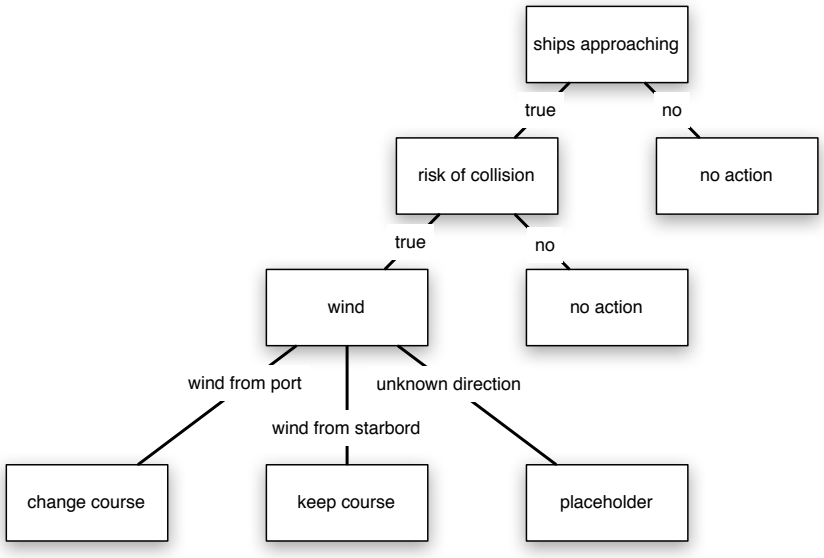


Figure 3.9 – Visualisation of agent behaviour hierarchy for equation 3.22.

Integration of the primitives for behaviour description is done by adding a check for correct behaviour, for example, defining a predicate that encodes "*has a course correction (movement) been initiated by the ship that was required to avoid the other?*". This enables very intuitive combinations of informal rules description checks with common-sense-based assumptions of agent actions, readily understandable for non-computer experts from different domains. The agglomeration of found behaviour description instances creates the high-level analysis of the domain.

3.8 Analysis Component

The analysis component is built on evaluating all instances found in the matching step. The results are all description predicate instances which can be fulfilled by the provided observations. Collecting instances of the described agent behaviours generates a high-level report on what occurred in a scenario. Matched agent behaviour instances carry the start and end times of the behaviour instances themselves, as well as additional information such as spatial data.

The following four analysis categories were determined based on fields of interest expressed during the interview with the domain expert (Podebry, 2014).

Spatial collocation of events over a period of time Zones of interest are emphasised by aggregating found instances for behaviour descriptions in a given spatial region over time. Applied to the running example from the navigational domain, it is possible to mark zones in which interactions occur between two ships, as specified by the COLREGS. In Figure 3.10, a heat map visualises zones with different levels of rule activations over three days based on the data sets used in the evaluation (see Section 5.2). For reference, the heat map overlays the visualisation of ship trajectories in that region, recognisable as yellow lines. Human operators may notice the congruence of zones with higher rule activations and sea navigation corridors, particularly when intersected with traffic from the ports of Dover and Calais. Another active hotspot is located near the port on the right side of the map, as indicated by the bright yellow colour. This combination of high-level scenario analysis with visualisation allows a more intuitive understanding of the dynamic scenario than a collection of numbers in tables.

By statistically comparing the numbers of collected instances against participating agents, known features of the observed domain, such as high-risk areas (large number of dangerous situations or similar features), or previously assumed hypotheses of the scenario can be verified or falsified.

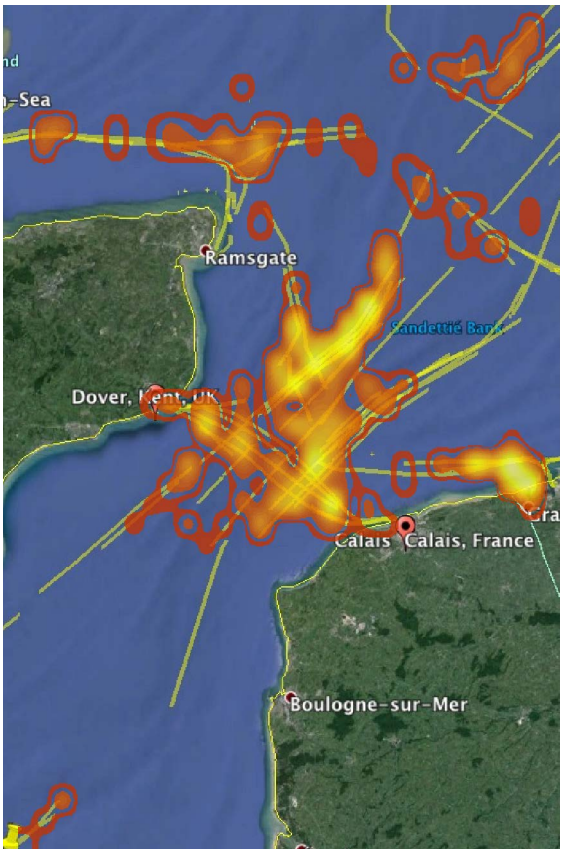


Figure 3.10 – Derived heat map for 2-dimensional histogram of found COLREG instances for naval traffic (background map copyright by Google Inc.).

Temporal collocation of events within a given spatial region An histogram visualises the distribution of occurrences of behaviours purely over time, as seen in Diagram 3.8.1. The histogram is especially well suited for highlighting temporal patterns of elevated levels of instances or their absence. Both features indicate noteworthy aspects for the overall analysis, allowing, for instance, the identification of areas that need a higher level of resources (for example, the personnel needed to manage certain traffic routes) or times where the work load can be reduced based on a low frequency of instances.

In-depth analysis Once the broader overview of events in the observed scenario is established, the analyst is able to traverse instances of selected groups of agents or sole agents to evaluate their behaviour. The collection of behaviour instances allows for behaviour descriptions down to the level of lone-acting instances. An example is irregular behaviour for a ship that is recorded as repeatedly not following the COLREGS

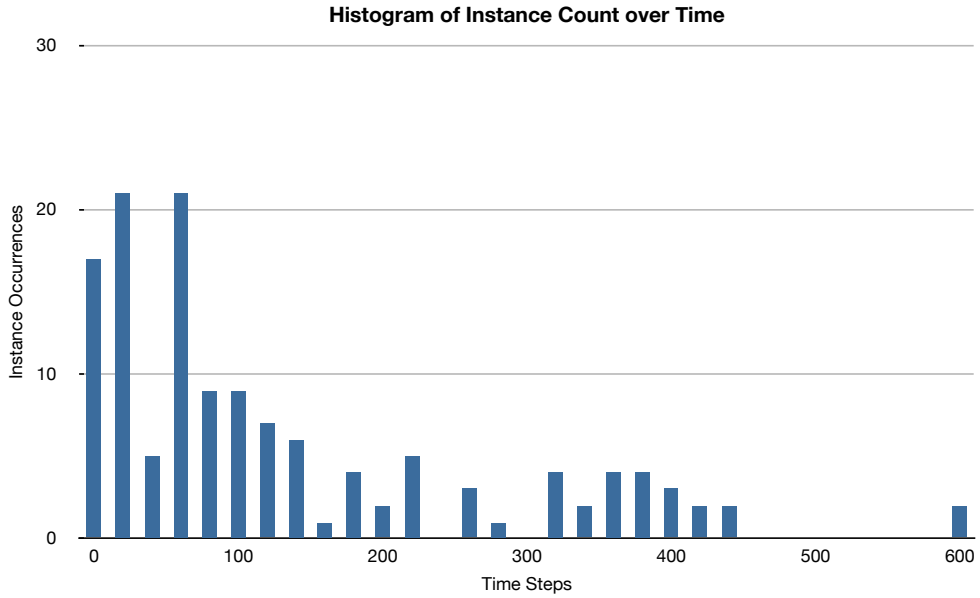


Diagram 3.8.1 – Histogram of behaviour instances over time for the Dover - Calais example.

specifications. Such behaviour is detected by combining found instances of spatial configurations that fulfil the conditions for the formalised rules and a check for the following rule-compliant execution, either holding its course or making an avoidance movement as specified in Section 4.5.1.

Absent Instances Noting absent agent behaviour matches or those that have surprisingly low matching rates hints at the possibility of either missing observational data or incorrect behaviour descriptions. The former is supported if there are gaps in the temporal or spatial information of the matched behaviour instances. The latter is more probable if no gaps in the observational data are obvious and must be handled in cooperation with domain experts.

Examples of blind spots can be found in the lower regions of Figure 3.10. A probable explanation is the absence of moving agents in that area. Another example is found in Diagram 3.8.1 in the time steps ranging from 450 to nearly 600. The absence of rule instances found there hint at either a pause in maritime movements or incomplete observation data during that time.

3.9 Chapter Summary

This chapter began with the description of the notation used. The spatial temporal predicates that encapsulate the $\mathcal{OPRA}_m$ representation and the Linear Temporal Logic relations were introduced and distinct predicates to handle relations with time and predicates for spatial relations were constructed. By combining the two concepts, predicates for turns and movements were be created.

After the predicates were defined, two methods for the generation of hypothetical facts were described. Such hypothetical facts are used to close gaps in the observational data sets or to give the analytical information about unobserved facts that are vital to the understanding of the scenario. This thesis focuses on enhancing the analytical properties of the system presented and will therefore feature the abductive hypothetical fact generation method in favour of the proximity based, which focuses on enhancing the percentage of found partially observed agent behaviours.

The chapter is closed with a description of how the usual method for finding instances is realised.

Chapter 4

Application to Maritime Navigation

Navigation in maritime environments is a domain well-suited for testing the usefulness of the introduced methods, due to the availability of clearly structured agent behaviour descriptions in the form of the Convention on the International Regulations for Preventing Collisions at Sea, 1972 (COLREGs), see Appendix A. The benefits in this domain are manifold. Analysis of observable agent behaviour descriptions results in an enhanced overall understanding of the current situation by the personnel responsible for steering sea vessels. Automated systems can also be integrated to highlight unexplainable[1] situations or even rule-breaking behaviour detected by the analysis. Additionally, by assuming goal-driven behaviour of the naval vessels, hypothetical facts, e.g., unseen vessels, can be generated to explain non-default behaviours.

In this chapter, I focus on the recognition of formalised agent behaviour in real-world data. The present effort is similar to Kreutzmann, Wolter, Dylla and Lee (2013), who explored another possible method of modelling the rules using qualitative representations. I go beyond that previous work by complementing my novel modelling approach with a focus on handling large sets of real-world data. The formalisation of COLREGS aims at utilisation of the generally applicable qualitative description primitives introduced in Chapter 3. In detail, this chapter contains:

- details of observational data available in the domain and its formalisation.
- incorporation of background knowledge used for analysis and its formalisation.
- descriptions of domain rules codifying the available agent behaviour descriptions and their formalisation in Answer Set Programming.

[1] based on the naval rule set.

- insertion of hypothetical facts based on behaviour descriptions.
- application of the analysis method.

The chapter begins with an explanation of input data and its abstraction, defines background knowledge, and shows the conversion of expert knowledge into its formal representation in a notation that conforms to the syntax of Answer Set Programming.

4.1 Sensor Data Characteristics

Real-world data about ship movements is recorded and broadcasted via the Automatic Identification System (AIS) (Harati-Mokhtari et al., 2007) (Figure 4.1 and 4.2), a radio-based communication system that broadcasts Maritime Mobile Service Identity (MMSI), position, heading, and velocity, as well as a range of optional information like cargo, tonnage, destination, etc.. Both mandatory and optional data are sent in the National Marine Electronics Association (NMEA) format.

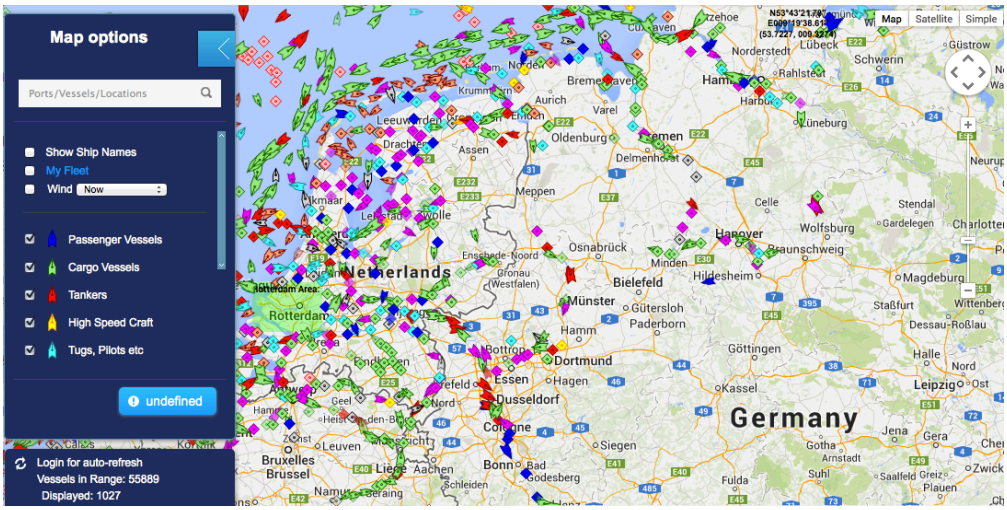


Figure 4.1 – Screenshot from `http://www.marinetraffic.com/` on December 3, 2013, showing northern Germany and the Netherlands overlaid with AIS data. An arrow indicates movement, a diamond shape a stationary vessel.

The AIS system was originally promoted by the International Maritime Organisation (IMO) to enhance safety and efficiency of navigation, safety of life at sea, and improved situational awareness (International Maritime Organization (IMO), 2003). It is now a mandatory system for commercial vessels to reduce traffic accidents, especially in bad weather conditions.

It also has drawbacks. As the information broadcasts are via VHF radio frequency, signals may be blocked due to interference. In the case of multiple vessels broadcasting at the same time in overlapping radio bands, only the strongest signal will be readable, if any. Various attempts to improve the system have been made, but so far no automated process recognition approaches have been applied (Hasegawa et al., 2008; Merchant et al., 2012). As noted in (Harati-Mokhtari et al., 2007), poor performance and the transmission of erroneous information by AIS are important issues that can affect its usefulness and were raised in the 16th session of the IALA AIS Committee (Sandford, 2005).

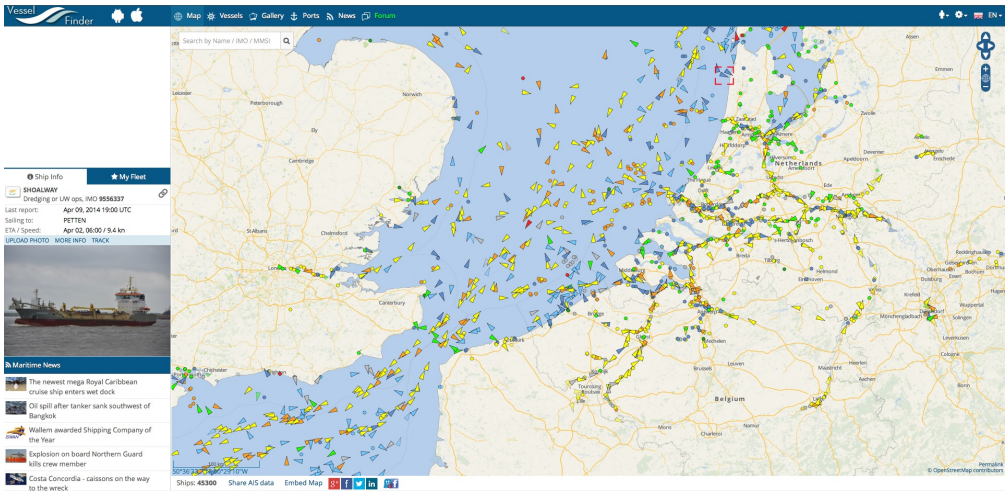


Figure 4.2 – Screenshot from AISHub `http://www.vesselfinder.com/` on April 9, 2014, southern England and part of France. The legend for the displayed entities is identical to Figure 4.1.

To remain as faithful as possible to real-world conditions, the formalised datasets for this work incorporate actual AIS information density and faultiness. Derived from the NMEA data format, the following base facts are used in the instantiation of the agent behaviour descriptions:

- Identification as
  $vessel(shipname)$ primitives.
- Cardinal orientation of the ships as
  $heading(time, identification, cardinal\ direction)$.
- Positional data, abstracted from GPS notation to planar x, y coordinates. Spherical projection is unnecessary because relations are analysed only between relatively close entities.

Additionally, the following information is derived from the collected AIS data. Given the positional data, distances are computed and compared against a threshold for collision danger. Also, as global headings are known, relative orientations can be determined.

- Binary qualitative distance information in the form of $closeEnoughToCollide(time, shipname_i, shipname_j)$
- Orientation relative to other entities as an $\mathcal{OPRA}_{m\ =4}$ predicate $opra(time, shipname_i, shipname_j, oprapredicate_l, oprapredicate_m)$

Heading information, usually broadcast as angular information, is discretised into a qualitative representation, compare Figure 4.3. For sailing ships, wind is an essential element in the collision-avoidance rules. As weather conditions are available from forecasts, measured by the ships themselves, or recorded for later use, it is reasonable to assume the information can be obtained by ships in real time.

- Global wind direction as an $\mathcal{OPRA}_m$ predicate $opra(time, shipname_i, wind, oprasamepredicate, opra\ predicate)$.

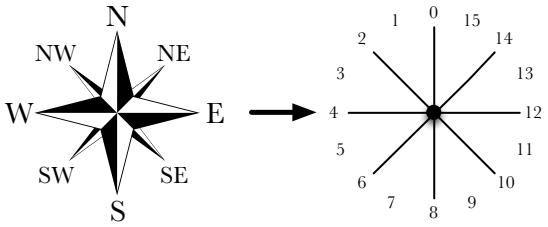


Figure 4.3 – Discretisation from cardinal direction to a qualitative notation. The numbering on the right-side representation marks lines as well as cone segments.

I represent the movement of a sea vessel over time as a list of orientated points, as depicted in Figure 4.4. This creates a framework for applying the reasoning and analysis methods described in Chapter 3.

The introduced predicates are already in a syntax that can be used by ASP-conforming solvers. To correctly interpret the sensor measurements in the AIS data also requires common-sense knowledge and environmental knowledge about entities and phenomena in the domain. Furthermore, descriptions are needed of the agent behaviours occurring in the domain. Both are handled in the following sections.

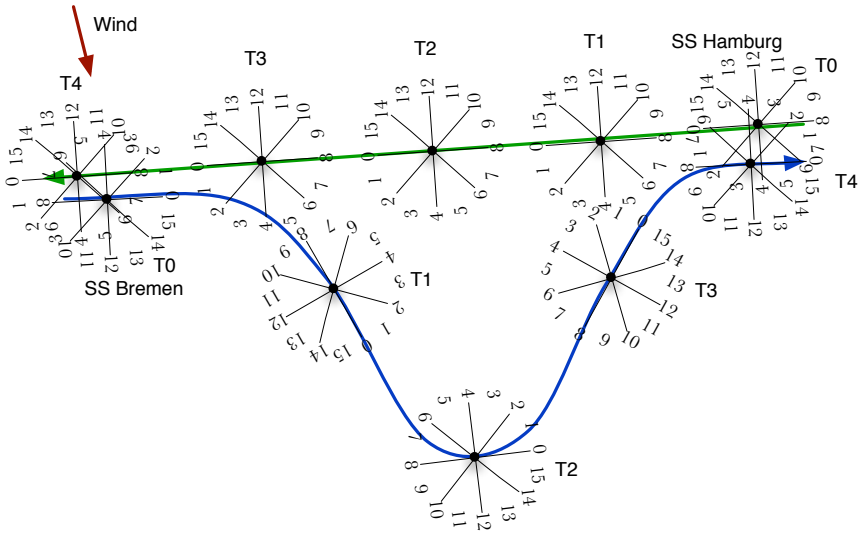


Figure 4.4 – Collision avoidance with $\mathcal{OPRA}_m$ predicates. The figure depicts the movement of two ships over 5 time steps ($T_0..T_4$). The $SS\ Bremen$ (blue line) is avoiding the $SS\ Hamburg$ (green line) according to COLREGS, § 12 (a) (ii).

4.2 Background Knowledge

Two pieces of knowledge significant to sea navigation are included in the reasoning process. First is the fact that wind significantly affects part of the sea vessels, i.e., sailing boats[2], and is referenced by the COLREGS.

4.2.1 Wind

Wind predicates are relative to an $entity$ and are described as 'the direction from which the wind impacts the entity ($opradirection$)', as well as its $cardinal\ direction$. Wind direction can change over time, so the wind predicate has an associated time stamp to ensure the correct wind direction for each time point.

$$opra(timepoint_i, entity_j, wind, cardinaldirection_k, opradirection_l) \forall i,j,k,l \in \mathbb{N}. \quad (4.1)$$

[2]All entities above water level are affected by the wind, but for reasons of simplicity, as well as with regard to the modelling of the COLREGS, this work limits the effect to sailing vessels.

4.2.2 Common Sense

Secondly, it is assumed that each vessel on the sea has a $goal$ or local target towards which it moves. This greatly influences the reasoning process, as it helps define 'abnormal' movements that deviate from the previous path[3]. Note that there is no need for knowledge about the goal itself, as assuming that the vessel has a preferred direction suffices for the definition of a path deviation. When assuming goal-directed motion without further knowledge about the environment, it is necessary to exclude vessels that move along coastlines or waterways, for example in the "Wattenmeer" close to Germany's northwestern coastline. This constraint can be overcome by adding environmental knowledge for comparison to vessel trajectories, but that is not within the scope of the thesis because of its minor importance in demonstrating that the detection of 'abnormal' movements is at all possible in qualitative terms.

4.3 Expert Knowledge

The expert knowledge in this thesis is based on the COLREGS, see Appendix A. They have been composed and are regularly discussed by domain experts. I concentrate on the rules that concern spatial and temporal change, an overview of which is given in Table 5.1. This includes the rules that regulate the interaction of two sailing boats, and crossing as well as reciprocal and overtaking situations for power-driven vessels. The relevant parts for the COLREGS are located under Rules 12 to 15, see Section 4.3.1. Rules 16 to 19 also regulate movements in the broader sense, but are not relevant to this thesis. The following section cites the relevant parts, visualises them, and formalises spatiotemporal relations. The formalisation was reviewed by a domain expert (Christian Podebry (2014), currently[4] helmsman for AIDA cruises 'club resort' ships).

The angular configurations have been chosen to cover all distinguishable scenarios with respect to the COLREGS.

4.3.1 COLREG Rule 12: Protocol for interaction of two sailing vessels approaching one another

(a) When two sailing vessels are approaching one another, so as to involve risk of collision, one of them shall keep out of the way of the other as follows:

[3]There are NMEA messages for the indication of avoidance movements, but they seem to be used rarely, if at all, and were not been found in the data recorded for this thesis.

[4]April 22, 2014

(i) when each has the wind on a different side, the vessel which has the wind on the port side shall keep out of the way of the other, see Figure 4.5.

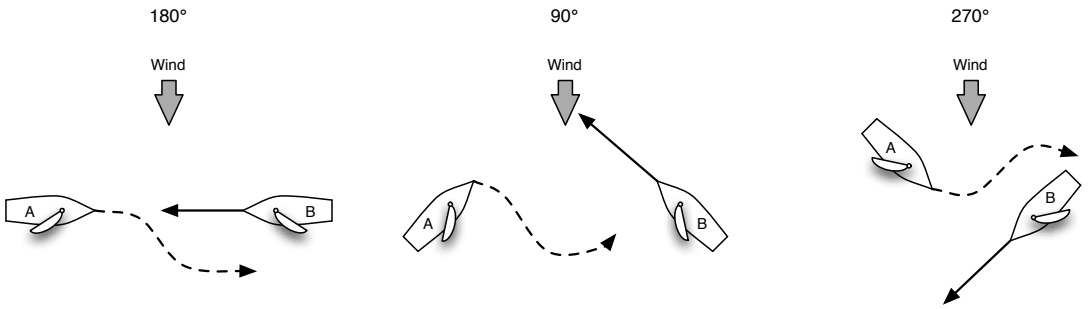


Figure 4.5 – Schematic view of three possible collision avoidance patterns for two sailing vessels. Here the ships have the wind coming from port side for vessel A and starboard for vessel B.

(ii) when both have the wind on the same side, the vessel which is to windward shall keep out of the way of the vessel which is to leeward, see Figure 4.6.

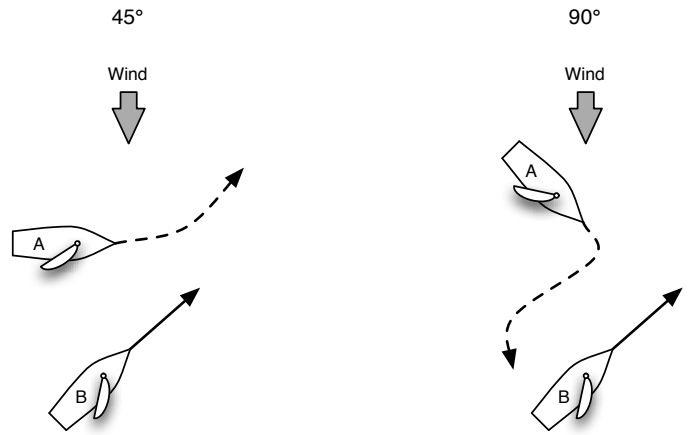


Figure 4.6 – Schematic view of two possible collision avoidance patterns for two sailing vessels. Here the ships have the wind coming from starboard for vessel A and vessel B.

(iii) if a vessel with the wind on the port side sees a vessel to windward and cannot determine with certainty whether the other vessel has the wind on the port or on the starboard side, she shall keep out of the way of the other, see Figure 4.7.

(b) For the purposes of this rule the windward side shall be deemed to be the side opposite to that on which the main sail is carried or, in the case of a square-rigged

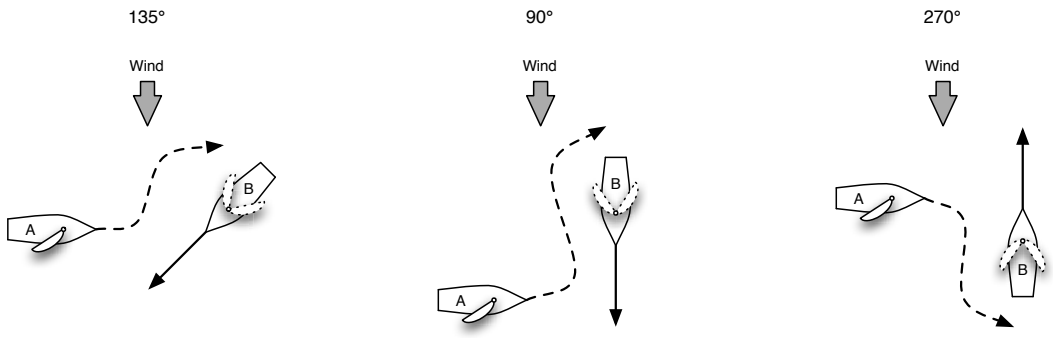


Figure 4.7 – Schematic view of three possible collision avoidance patterns for two sailing vessels. Here the first ship has the wind coming from port side and the wind direction for the second vessel is unclear.

vessel, the side opposite to that on which the largest fore-and-aft sail is carried.

Most interesting here are the parts (a) (i) and (ii) as they regulate which of two sailing vessels should stay on course in case of a possible collision and which should turn, compare Figures 4.5 and 4.6. The rules are already formulated qualitatively (Dylla, 2008) as sets of $\mathcal{OPRA}_m$ neighbourhood graph relations. This fact is made use of in the hypothesis generation section, but the encoding of avoidance rules is done based on the predicate structure from Section 3.4.1.

4.3.2 Formalised Collision Avoidance Rule 12, (a), (i)

The rule is split into two top-level predicates, $change\_course(T, X)$ and $keep\_course(T, X)$ to differentiate between the two relevant ships. Here, T is the indicator for the time point and X denotes the entity's identity.

First, a predicate is needed to state whether the ships are actually on a collision course and are close enough to collide. In the code element in Algorithm 1, this is achieved using the predicate $couldCollide(T, X, Y)$, with the additional variable Y, denoting the other ship, see Algorithm 3. It is based on two predicates, $closeEnoughToCollide(T, X, Y)$ and $opraOppositeDir(T, X, Y)$. The predicate $closeEnoughToCollide(T, X, Y)$ symbolises a threshold for safe movements and has to be chosen by domain experts depending on factors like tonnage, rate of turn, or similar, whereas the predicate $opraOppositeDir(T, X, Y)$ denotes a qualitative $\mathcal{OPRA}_m$ arrangement of two ships facing each other.

The $opraOppositeDir(T, X, Y)$ predicate is based on the spatial predicate $ahead(T, X, Y)$. It is true if both entities are ahead of each other.

```
change_course_rule12_i(T, X, Y) :-
        couldCollide(T, X, Y),
        windNotSameDir(T, X, Y),
        windDir(T, X, port),
        vessel( X, sailingboat ), vessel( Y, sailingboat ),
        not hasFiredPreviously(T, rule12_i, X, Y).

keep_course_rule12_i(T, X, Y) :-
        couldCollide(T, X, Y),
        windNotSameDir(T, X, Y),
        windDir(T, X, starboard),
        vessel( X, sailingboat ), vessel( Y, sailingboat ),
        not hasFiredPreviously(T, rule12_i, Y, X).
```

Algorithm 1: Collision Avoidance Rules 12, (a), (i)

```
couldCollide(T, X, Y) :-
        closeEnoughToCollide(T, X, Y),
        opraOppositeDir(T, X, Y).
```

Algorithm 2: $couldCollide(T, X, Y)$ predicate

```
opraOppositeDir(T, X, Y) :-
        ahead(T, X, Y), ahead(T, Y, X).

ahead(0..m/2).
ahead(7*m/2..4*m).
ahead(T, X, Y) :- opra(T, Y, X, R, _), ahead(R).
```

Algorithm 3: $opraOppositeDir(T, X, Y)$ predicate

The fourth and fifth lines of code in Algorithm 3 need mentioning, as they map the granularity of the $\mathcal{OPRA}_m$ predicate to $m = 4$ segmentation. As seen in Figure 3.4, this corresponds to a quarter of a circle in front of the entity.

Then, a predicate is needed which states the fact that both boats have the wind from different sides, $windNotSameDir(T, X, Y)$, Algorithm 4. Here I simply state that the wind direction is not permitted to be identical, as there are starboard and port as alternatives. The predicate $opra(T, X, wind, s, Z)$ maps directly to the observed data in the form of an $\mathcal{OPRA}_m$ self predicate.

```
1  windNotSameDir(T, X, Y) :- windDir(T, X, Z), windDir(T, Y, Z2), Z != Z2.
2
3  windDir(T, X, port) :- opra(T, X, wind, s, Z), opraPort(Z).
4  windDir(T, X, starboard) :- opra(T, X, wind, s, Z), opraStarboard(Z).
```

Algorithm 4: $windNotSameDir(T, X, Y)$ predicate

Important here is the mapping from the domain-dependent $opraPort(Z)$ and $opraStarboard(Z)$ predicates to domain-independent direction predicates ($opraLeft(Z)$, $opraRight(Z)$) for a visualisation of the $\mathcal{OPRA}_m$ values, see Figure 3.6 and 3.7. Both predicates are egocentric definitions from a point.

```
1  opraLeft(Z) :- Z = 0..(2*m-1).
2  opraRight(Z) :- Z = (2*m)..(4*m-1).
3
4  opraPort(Z) :- opraLeft(Z).
5  opraStarboard(Z) :- opraRight(Z).
```

Algorithm 5: $opraPort(Z)$ and $opraStarboard(Z)$ predicate

Analogous to that, I define the predicate that denotes the identical wind direction, Algorithm 6.

```
1  windSameDir(T, X, Y) :- windDir(T, X, Z), windDir(T, Y, Z), X != Y.
```

Algorithm 6: $windSameDir(T, X, Y)$ predicate

Additionally, the code checks whether a vessel confirms to the correct type with the $vessel(X, sailingship)$ predicate, as in code segment 7.

```
vessel( X, sailingboat )
```

Algorithm 7: $vessel$ predicate

And finally, further rule invocation is blocked with $nothasFiredPreviously(T, rule12_i)$. The predicate allows only the first fitting pattern to be matched, compare code block 8. Here, to count the rule as fired requires two vessels, one that changes its course and one that keeps it. Similar blocking rules have been written for all further rules and can be found in the attached code but are not explicitly explained for each.

```
ruleFired(T, rule12_i, X, Y) :-
        change_course_rule12_i(T, X, Y),
        keep_course_rule12_i(T, Y, X).

hasFiredPreviously(T, RN, X, Y) :-
        timepoint(T),
        timepoint(T2), T2 < T,
        ruleFired(T2, RN, X, Y).
```

Algorithm 8: $ruleFired$ predicate

The last predicate in the top-level Algorithm 1 of the rule formalisation is the decision regarding which ship should keep its course and which should turn, according to rule 12, (a), (i).

4.3.3 Formalised Collision Avoidance Rule 12, (a), (ii)

I follow with the second part of Rule 12:

Here, the $ahead(T, X, Y)$ predicate from equation 3.12 in Chapter 3.4.3 is used to ensure that the ships are steering towards each other. Also, I introduce the new predicate $inLuvOf(T, X, Y)$ and $inLeeOf(T, X, Y)$. The predicates Luv and Lee come from the naval domain itself and denote the relation of one ship with respect to the wind direction. Luv denotes the side of the ship that receives the wind, whereas Lee is the side that does not have wind incoming, see Figure 4.8. Other than these two predicates, the encoding remains the same.

```
change_course_rule12_ii(T, X, Y) :-
        ahead(T, X, Y),
        windSameDir(T, X, Y),
        inLuvOf(T, X, Y),
        vessel( X, sailingboat ), vessel( Y, sailingboat ),
        not hasFiredPreviously(T, rule12_ii, X, Y).

keep_course_rule12_ii(T, X, Y) :-
        ahead(T, Y, X),
        windSameDir(T, X, Y),
        inLee(T, X, Y),
        vessel( X, sailingboat ), vessel( Y, sailingboat ),
        not hasFiredPreviously(T, rule12_ii, Y, X).
```

Algorithm 9: Collision Avoidance Rules 12, (a), (ii)

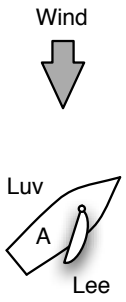


Figure 4.8 – Luv and Lee sides of a ship A.

4.3.4 Formalised Collision Avoidance Rule 12, (a), (iii)

I end with the third part of Rule 12. Here, all knowledge about (wind) conditions from the perspective of the other ship is omitted and replaced by the $notclearWindDir(T, X, Y)$ predicate, Algorithm 11.

In contrast to the previous rules, nothing about the behaviour of the second ship is specified in the COLREGS, so the $keep\_course$ part of the rules is omitted here.

Note the need to specify the types of the variables X and Y in all rules to avoid grounding problems[5]. Further, I specify that the two ships in the predicate may not be identical and that T is a time point. Everything else is accomplished by stating that there

[5]so-called 'unsafe variables' are unbound atoms found during the process of grounding. Each variable must be in the scope of an atom (in the body of the rule) that can bind it.

```
change_course_rule12_iii(T, X) :-
        windDir(T, X, port),
        inLuvOf(T, X, Y),
        notclearWindDir(T, X, Y),
        vessel( X, sailingboat ), vessel( Y, sailingboat ),
        not hasFiredPreviously(T, rule12_iii, X).
```

Algorithm 10: Collision Avoidance Rules 12, (a), (iii)

is no predicate defining the wind direction for the second ship. Here, there is also no need to specify which ship should keep its course, as no such requirement is specified in the COLREGS.

```
notclearWindDir(T, X, Y) :-
        vessel(X, sailingboat ),
        vessel(Y, sailingboat ),
        X!=Y,
        timepoint(T),
        not windDir(T, Y, starboard),
        not windDir(T, Y, port).
```

Algorithm 11: $notclearWindDir(T, X, Y)$ predicate

4.3.5 COLREG Rule 13: Protocol for interaction of two motor-driven vessels in the case of overtaking

(a) Notwithstanding anything contained in the rules of Part B, Sections I and II, any vessel overtaking any other shall keep out of the way of the vessel being overtaken.

(b) A vessel shall be deemed to be overtaking when coming up with another vessel from a direction more than 22.5[6] degrees abaft her beam, that is, in such a position with reference to the vessel she is overtaking, that at night she would be able to see only the stern light of that vessel but neither of her sidelights, see Figure 4.11.

(c) When a vessel is in any doubt as to whether she is overtaking another, she shall assume that this is the case and act accordingly.

[6]Note that the 22.5 degrees are $\frac{1}{16}$ of a full circle, adding up to $\frac{1}{8}$ for the derivation into each direction, fitting nicely into an $\mathcal{OPRA}_m$ with $m = 4$ representation.

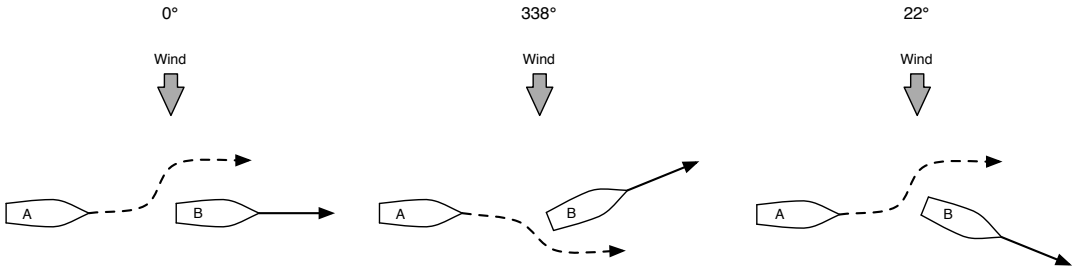


Figure 4.9 – Schematic view of three possible collision avoidance patterns for two vessels. The vessel abaft of the other is in a cone that does not deviate more than 22.5 degrees.

(d) Any subsequent alteration of the bearing between the two vessels shall not make the overtaking vessel a crossing vessel within the meaning of these Rules or relieve her of the duty of keeping clear of the overtaken vessel until she is finally past and clear.

Note that the rules are provided here for power-driven vessels. The rule itself makes use of a relative orientation between two entities.

4.3.6 Formalised Collision Avoidance Rule 13 (b)

Collision Avoidance Rule 13 describes the behaviour two power-driven vessels must exhibit when one overtakes the other. Here, overtaking is formalised as having two vessels close enough together that a collision is possible ($closeEnoughToCollide(T, X, Y)$ predicate as described in subsection 4.3.2) and making use of the $back(T, X, Y)$ primitive.

The $back(T, X, Y)$ primitive as depicted in Figure 3.5 is constructed analogously to $ahead(T, X, Y)$ by denoting the spatial configuration of two points. The second point is orientated in the cone spanning two sections of an $\mathcal{OPRA}_m$ segmented space on the back side of the first oriented point.

4.3.7 COLREG Rule 14: Protocol for interaction of two motor-driven vessels in the case of a head-on situation

(a) When two power-driven vessels are meeting on reciprocal or nearly reciprocal courses so as to involve risk of collision, each shall alter her course to starboard so that each shall pass on the port side of the other.

(b) Such a situation shall be deemed to exist when a vessel sees the other ahead or nearly ahead and by night she could see the masthead lights of the other in a line or nearly

```
change_course_rule13_b(T, X, Y) :-
    closeEnoughToCollide(T, X, Y),
    back(T, Y, X),
    ahead(T, X, Y),
    vesselPowerDriven( X), vesselPowerDriven( Y),
    not hasFiredPreviously(T, rule13_b, X, Y).

keep_course_rule13_b(T, X, Y) :-
    closeEnoughToCollide(T, Y, X),
    back(T, X, Y),
    ahead(T, Y, X),
    vesselPowerDriven( X), vesselPowerDriven( Y),
    not hasFiredPreviously(T, rule13_b, Y, X).
```

Algorithm 12: Collision Avoidance Rule 13

```
back(((3*m)/2)..((5*m)/2)).
back(T, X, Y) :- opra(T, Y, X, R, _), back(R).
```

Algorithm 13: $back(T, X, Y)$ primitive.

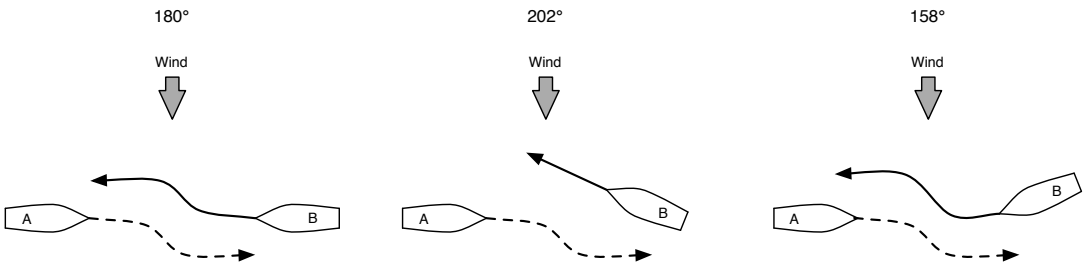


Figure 4.10 – Schematic view of three possible collision avoidance patterns for two vessels. The vessel abaft of the other is in a cone that does not deviate more than 22.5 degrees.

in a line and/or both sidelights and by day she observes the corresponding aspect of the other vessel.

(c) When a vessel is in any doubt as to whether such a situation exists she shall assume that it does exist and act accordingly.

4.3.8 Formalised Collision Avoidance Rule 14 (a)

Both of the power-driven vessels heading towards each other should take steps to avoid a collision. Based on the previously introduced predicates, the formalisation can be directly defined by use of the $couldCollide(T, X, Y)$ predicate which already encapsulates the requirements. Then it is only necessary to guarantee the correct vessel type.

```
1  change_course_rule14_a(T, X, Y) :-
2      couldCollide(T, X, Y),
3      vesselPowerDriven( X), vesselPowerDriven( Y),
4      not hasFiredPreviously(T, rule14_a, X, Y).
```

Algorithm 14: Collision Avoidance Rule 14

The $ruleFired()$ predicate is slightly changed for Rule 14 (a) by using two Rule 14 (a) predicates for which the acting ship agents have been swapped, see Algorithm 15. Both agents are required to be engaged in the rule.

```
1  ruleFired(T, rule14_a, X, Y) :-
2      change_course_rule14_a(T, X, Y),
3      change_course_rule14_a(T, Y, X), X!=Y.
```

Algorithm 15: Collision Avoidance Rule 14, rule check

4.3.9 COLREG Rule 15: Protocol for interaction of two motor-driven vessels in the case of one crossing the other

When two power-driven vessels are crossing so as to involve risk of collision, the vessel which has the other on her own starboard side shall keep out of the way and shall, if the circumstances of the case admit, avoid crossing ahead of the other vessel.

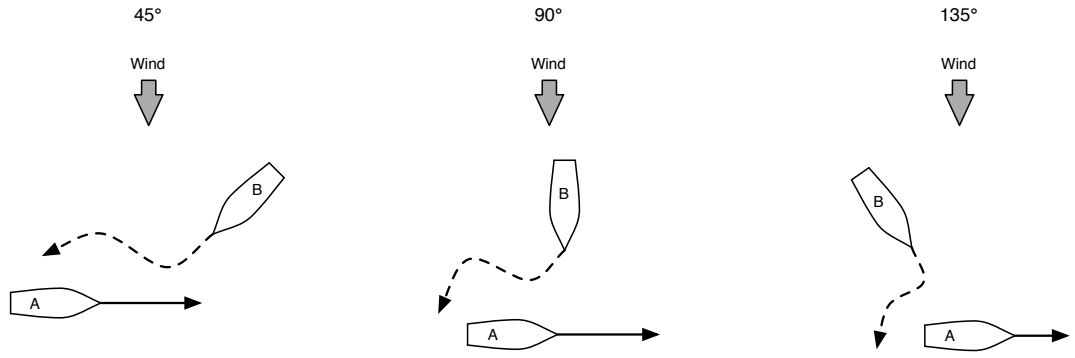


Figure 4.11 – Schematic view of three possible collision avoidance patterns for two vessels. The vessel abaft of the other is in a cone that does not deviate more than 45 degrees.

4.3.10 Formalised Collision Avoidance Rule 15

For the situation in which one power-driven vessel is crossing the other, the COLREGs specify that the vessel having the other on its starboard side should evade and if possible cross behind it. The latter optional requirement is not encapsulated in the formalisation due to the action taking place in the future. Here, I search for closeness of the two vessels and whether the avoiding vessel has the other on its starboard side, see Algorithm 16.

4.3.11 Summary: Options provided by Formalised Expert Knowledge

With formalised rules from Section 4.3 as the agent behaviour descriptions, it is now possible to compare the observed data to the rules. I make use of middle-level predicates

```
change_course_rule15(T, X, Y) :-
    closeEnoughToCollide(T, X, Y),
    not back(T, X, Y),
    onSide(T, Y, X, starboard),
    vesselPowerDriven( X), vesselPowerDriven( Y),
    not hasFiredPreviously(T, rule15, X, Y).
```

Algorithm 16: Collision Avoidance Rule 15

from Section 3.4.2 and 3.4.3 to simplify predicate generation. The domain predicates are directly derived from the easily formalisable rules of collision avoidance. To encode the rules, relative orientations of the ships involved and the direction of the wind are relevant. The relative orientation can be directly encoded in $\mathcal{OPRA}_m$ predicates that are constructed by the combination of cardinal directions of the ships. Wind directions can be assumed to be known as there are many weather reports available at all times[7].

4.4 Verification of Rule-Compliant Behaviour

In addition to the intuitive formalisation of the COLREGS, the qualitative analysis framework allows assessment of whether the ships involved in an engagement acted in compliance with the given behaviour rules. Based on the descriptions in the COLREGS, one or both involved ships have to make a course adjustment to avoid a possible collision. Each compliance test for a found behaviour-rule instance is similar, see Algorithm 17.

```
rule12_i_correct_executed(T2, X, Y) :-
        change_course_rule12_i(T, X, Y),
        turn(X, _, _, T2, _), T2 > T,
        closeEnoughToCollide(T2, X, Y).
```

Algorithm 17: Compliance check for COLREG rule 12 (a) (i). Rules 12 (a) (ii) to 15 are handled similarly.

The compliance check is initiated with an instantiation of a behaviour description. It is always examined from the perspective of one involved agent, in this case from the agent required to make the avoidance movement. The agent's behaviour is checked for

[7] As examples: `http://passageweather.com/`, `http://www.metoffice.gov.uk/weather/marine/`, or `http://www.nws.noaa.gov/om/marine/home.htm` (all last visited March 5, 2014).

the required turn (predicate $turn()$, compare Algorithm 18) at a later point in time, while still in the vicinity of the other involved ship agent ($closeEnoughToCollide()$ predicate) to ensure that movements belonging to later course corrections are not considered.

```
1  turn(X, HEADING1, HEADING2, T1, T2) :-
2      fact(T1, X, _, HEADING1),
3      fact(T2, X, _, HEADING2),
4      HEADING1 != HEADING2,
5      next(T1, T2).
```

Algorithm 18: Turn predicate

Notable differences in the compliance checks in Algorithm 17 are the vessel-type check for Rule 12 (a) (iii) to ensure correct interpretation.

4.5 Insertion of Hypothetical Facts

If only data about one ship is present but the observed vessel is behaving in an unexpected way, it is possible to match its behaviour to known avoidance patterns and generate hypotheses about possible ghost-ship positions.

4.5.1 Prerequisites for Insertion of Hypothetical Facts: Avoidance Detection

To enable hypothetical reasoning, I assume open-sea vessels follow a course with a heading towards their goal or are following predefined waterways.

```
1  avoidance(X, T1, T4) :-
2      turn(X, HEADING1, HEADING2, T1, T2),
3      turn(X, HEADING3, HEADING4, T2, T4),
4      HEADING1 = HEADING4,
5      after(T1, T4).
```

Algorithm 19: Detection of avoidance patterns.

This allows me to define movements that encode a turn starboard or port side and, at a later point in time, a turn back toward the previous direction, resulting in an avoidance

behaviour; compare Algorithm 21. Versions for starboard and port sides are implemented, as well as a more general avoidance predicate based on avoidance in either direction.

```
avoidanceStarboard(X, HEADING1, HEADING2, T1, T2) :-
    turnRight(X, HEADING1, HEADING2, T1, T2),
    turnLeft(X, HEADING3, HEADING4, T3, T4),
    HEADING1 = HEADING4,
    after(T1, T4).

avoidanceBackboard(X, HEADING1, HEADING2, T1, T2) :-
    turnLeft(X, HEADING1, HEADING2, T1, T2),
    turnRight(X, HEADING3, HEADING4, T3, T4),
    HEADING1 = HEADING4,
    after(T1, T4).
```

Algorithm 20: Detection of directed avoidance patterns.

Besides the introduced $turn()$ predicate, this algorithm ensures that the ship in question has returned to its previous course. Additionally, the ordering of the turns is set to differentiate the two avoidance manoeuvres, compare Algorithm 20. Not considered here is the manoeuvring of a sailing ship against the wind. Careful modelling of the domain can allow for it, however. Either a reoccurrence can be excluded from the avoidance detection or the avoidance can be limited to a smaller or even wider turn radius.

4.5.2 Hypothetical Fact Generation: Ghost Ship Cone Prediction

With knowledge about abnormal movements, i.e., avoidances without observed reasons, it becomes possible to generate hypothetical facts of unseen vessels using abductive reasoning and the integration of another set of predicates. Avoidance is assumed when the course of a ship deviates more than a $1/8$ segment of a circle.

By integrating a subpart of the formalised ship movements from the work of Dylla (2008), it becomes possible to guess where the unseen vessel should be. Additional required knowledge is the starting conditions of movement sets for two vessels engaged in the enactment of a COLREG rule. Dylla constructed all possible qualitative graph structures that are the neighbourhood-based transfers of configurations between vessels following said rules. In Algorithm 21 and 21 these are named $BEHAVIOUR$. As these graphs need an initial setting, I use these to infer the position in qualitative spatial terms

```
ghostshipInOpraBack(TSTART, X, OPRA1, OPRA2, BEHAVIOR,
        VESSELTYPEOTHER, XORIG, YORIG ) :-
    avoidanceBackboard(X, HEADING1, HEADING2, T2, T2),
    behavior(BEHAVIOR),
    types(BEHAVIOR, VESSELTYPE, VESSELTYPEOTHER),
    vessel(X, VESSELTYPENEW),
    vesselTranslate(T1, X, VESSELTYPENEW, VESSELTYPE),
    startConditions( BEHAVIOR , OPRA1, OPRA2 ),
    global_position( T1, X, XORIG, YORIG, _ ),
    TSTART = T1 - 1.
```

Algorithm 21: Detection of avoidance patterns, port

of the unseen vessel if the observed vessel performs an avoidance movement. Position, in spatial-temporal terms for prediction of the position of an unseen vessel, refers to the means use in the representation, namely that a direction is cone shaped, outgoing from the position of the observer. The maximum range of the possible position in that cone can be limited by applying the threshold for the distance between two ships in the COLREGs.

```
ghostshipInOpraStar(TSTART, X, OPRA1, OPRA2, BEHAVIOR,
        VESSELTYPEOTHER, XORIG, YORIG ) :-
    avoidanceStarboard(X, HEADING1, HEADING2, T1, T2),
    behavior(BEHAVIOR),
    types(BEHAVIOR, VESSELTYPE, VESSELTYPEOTHER),
    vessel(X, VESSELTYPENEW),
    vesselTranslate(T1, X, VESSELTYPENEW, VESSELTYPE),
    startConditions( BEHAVIOR , OPRA1, OPRA2 ),
    global_position( T1, X, XORIG, YORIG, _),
```

Algorithm 22: Detection of avoidance patterns, starboard

To satisfy all conditions for the generation of a cone for a ghost ship[8] requires an avoidance movement from section 5.5. Further, it requires that the variable $BEHAVIOR$ is defined as such. The use of the $types()$ predicate reduces the possible projection cones based on the type of vessel of the avoiding ship, as not all behaviours fit all vessels. The predicate $vesselTranslate()$ is a conversion from the introduced naming conventions for naval vessels to those in Dylla's work. The predicate $startConditions()$ is directly

[8] So-called because it has not been seen.

extracted from the neighbourhood-based graph formalisations and defines all possible $\mathcal{OPRA}_m$ relations that are valid as a start condition for manoeuvres.

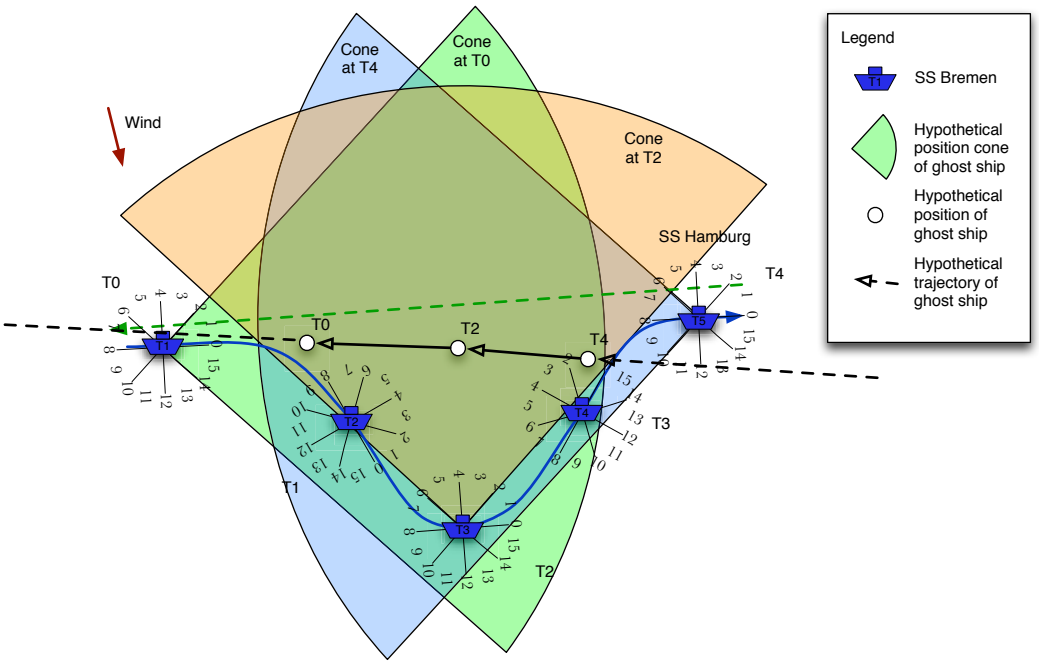


Figure 4.12 – In comparison with Figure 4.4, this figure depicts the movement of one ship over five time steps. Coloured cones indicate possible positions for the unseen ship. The black line with white dots is one possible interpretation of the trajectory of the unseen ship.

Figure 4.12 displays a prototypical cone projection for a vessel that avoided another vessel. At different time points, based on the formalised agent behaviour description, it becomes possible to extract a potential position cone for the other vessel by traversing through the graph structure of possible $\mathcal{OPRA}_m$ configurations. As this can be done for every time point that holds a discernible qualitative difference in the description (like a change in cardinal direction resulting in transfer from one part of the qualitative neighbourhood-based graph to the next), trajectories, albeit highly speculative ones, can be formed for ghost ships.

As seen in the example in Figure 4.12, the blue ship SS Bremen is coming from the left at a time point $T0$ on its default course. The ship starts an avoidance right. With the knowledge of that avoidance, acquired after the manoeuvre is completed and projected backwards in time, it is inferable that there was likely a ghost ship. What is more, it is

inferable that the ghost ship had to be in front of the SS Bremen at time point $T0$, at time point $T2$ to the left of the SS Bremen, and at time point $T4$ has been passed and is thus behind the SS Bremen. Based on the predicted cones, points where the ghost ship had been can be inferred and the large mass of possible positions constricted by the assumption that the ship had to hold its course according to the COLREGS. This yields a set of possible positions $\mathcal{P}_{\mathcal{POS}}$ for each time step in the set of time points $\mathcal{T}$ with a qualitative change in description of the avoidance (each $cone$), and the requirement of a straight line on which the points have to be placed to form a $trajectory$:

$$trajectory_i = straitline(pos_0, \ldots, pos_t) \text{ with } pos \in \mathcal{P}_{\mathcal{POS}}, i \in \mathbb{N} \text{ and } t \in \mathcal{T}.$$

4.6 Chapter Summary

This chapter included the formalisation of the domain rules, in this case the "Convention on the International Regulations for Preventing Collisions at Sea, 1972". The relevant rules concerning spatial-temporal relations between ships were made formal by applying the base predicates from the third chapter. In the case that specialised observational data was required, as for instance with the distance for the possibility of collisions, domain-dependent predicates have been inserted. After the sensor data format and the resulting predicates were described, a formalisation for the generation of hypothetical facts based on abductive reasoning was made.

Chapter 5

Evaluation

This chapter evaluates whether the proposed methods in this thesis (1) work correctly, (2) are applicable to real-world data, and (3) scale appropriately in terms of time and space. To this end, the first part of the chapter is dedicated to describing the coverage of domain rules, Section 5.1.1, and an evaluation of the correct detection of COLREGS-relevant spatiotemporal agent behaviours and their classification, Section 5.1.2. Then, synthetically generated scenarios are matched against the formalised naval rule sets and checked for validity.

Section 5.2 deals with the integration of real-world data. Data in the form of raw AIS signals are collected from a live data stream. Raw data is abstracted into qualitative descriptions, then matched against the formalised descriptions in Section 5.1.2. Here, it is demonstrated that the system is sufficiently robust to handle real data and is scalable when the data set grows.

This last element is thoroughly investigated in Section 5.3 by comparing the number of predicates generated and the time needed for solving the tasks. The final section shows how the system scales when confronted with large data sets.

5.1 Coverage and Validity

Two factors give insight into the quality of a formalisation of agent behaviour descriptions for a scenario. First, it must be verified that the formalised rules actually cover the behaviours that occur in the given domain. Second, the covered rules have to be correct and support the occurrences that can be observed.

5.1.1 Coverage of the Domain Rules

A vital factor for the effectiveness of functional agent behaviour descriptions in a domain is the ratio of the agent behaviours that occur in the domain compared to the formalised agent behaviours. In the ideal case, all occurring agent behaviours are modelled in a 1:1 ratio, fully representing the domain with the formal description.

Data Set for the Coverage Analysis

To express knowledge about the coverage of a given domain by the formalised rules, it is necessary to specify the set of all rules known for the domain $\mathcal{R}$ and the set of all formalised rules, called agent behaviour descriptions $\mathcal{ABD}$. Consider four different cases:

1. the set sizes of $\mathcal{R}$ and $\mathcal{ABD}$ are equal ($|\mathcal{R}| = |\mathcal{ABD}|$), resulting in a bijective mapping of all rules $r \rightarrow adb, \forall r \in \mathcal{R}, abd \in \mathcal{ABD}$. In this case, full transfer from the rules to the agent behaviours is assumed. This is the ideal case, which fully projects the real world onto the symbolic representation.

2. the set $\mathcal{R}$ is a subset of $\mathcal{ABD}$ ($\mathcal{R} \subset \mathcal{ABD}$). In this case, more formalised agent behaviours are generated than rules are found in the domain. This is typically the case when a surplus of agent behaviours is present. The world is subsequently over represented.

3. the set $\mathcal{R}$ is a superset of $\mathcal{ABD}$ ($\mathcal{R} \supset \mathcal{ABD}$). Here, it is reasonable to suppose that not all rules have been formalised and potential agent behaviours have been overlooked.

4. the set $\mathcal{R}$ has no overlap with the set $\mathcal{ABD}$ ($\mathcal{R} \cap \mathcal{ABD} = \varnothing$). In this case, the rules have nothing to do with the world. This should be avoided at all costs.

For the scenario at hand, I applied the formalisation framework developed earlier in this thesis to a subpart of the COLREGS. The COLREGS set forth various regulations for the seafaring domain, so the application scenario is limited to the rules concerning spatial-temporal relations of ships.

Results of the Coverage Analysis

The COLREGS are divided into 5 parts (Table 5.1), the first of which contains general definitions and information. In the second part, conventions for steering and sailing

COLREGS	Spatial relevance[1]	Temporal relevance[1]	Formalised	Topic
Part A				General rules
Rule 1-3	□	□	□	
Part B Section 1				Steering and Sailing
Rule 4	□	□	□	Section definition
Rule 5	□	□	□	Maintain proper lookout
Rule 6	✓	✓	□	Maintain safe speed
Rule 7	□	□	□	Warning for scanty information
Rule 8	□	□	□	Action to avoid collision (general)
Rule 9	✓	□	□	Action in narrow channels
Rule 10	✓	□	□	Near traffic separation schemes
Part B - Section 2				Conduct of vessels in sight
Rule 11	□	□	□	Section definition
Rule 12	✓	✓	✓	Collisions of sailing vessels
Rule 13	✓	✓	✓	Overtaking situations
Rule 14	✓	✓	✓	Head-on situations
Rule 15	✓	✓	✓	Crossing situations
Rule 16	□	□	□	Actions for the give-way vessel
Rule 17	✓	□	□	Action of the stand-on vessel
Rule 18	□	□	□	Responsibilities between vessel types
Part B - Section 3				Conduct of vessels in restricted visibility
Rule 19	□	□	□	Safe speed
Part C				Lights and Shapes
Rule 20-31	□	□	□	
Part D				Sound and Light Signals
Rule 32-37	□	□	□	
Part E				Exceptions
Rule 38	□	□	□	

Table 5.1 – COLREGS coverage. Rules formalised in this work are denoted by ✓, those not formalised by □.

are outlined; this is the section relevant[1] to rule formalisation. Parts C, D, and E cover lights and shapes, their use, and exceptions. The rules have been segmented with respect to their spatiotemporal relevance. Rules without spatial-temporal relevance are not modelled.

Three cases of collision avoidance in the COLREGs are relevant for sailing ships: Part B Section 2 Rule 12, compare Section 4.3.1. Three top-level agent behaviours were

[1] with respect to the spatial representation of this thesis.

generated for both of the vessels involved for the given scenario, compare Section 4.3.2 to 4.3.4 . This results in a bijective mapping of rules to agent behaviours per ship. Note that the evaluation of the domain coverage only considers top-level agent behaviour descriptions. For example, Rule 12 is separated into two parts, the first prescribing actions in the case of potential collision, followed by a clarification of the term "windward side". The first part is divided into 3 elements (i, ii, iii) that address different wind conditions. A rule is composed for each variant.

Rules 13, 14 and 15 handle collision avoidance for power-driven vessels. Similar to the rule sets for sailing boats, those agent behaviours have also been formalised.

Not formalised are the relevant spatial-temporal parts from rule 6, the only rule addressing both spatial and temporal concerns.

Discussion

Considering the array of spatial-temporal rules existing in the domain, this thesis thoroughly represents the spatial-temporal subdomain with the exception of checks for velocity. More precisely, this thesis deals with all cases that include relative-orientation descriptions for the given domain.

Rule 6 does not concern point-orientated relations for two entities. Instead, to measure velocity accurately would require at least three points and a fine granular model. Although it is possible to estimate velocity using an integration of measurements over time, the possible error margin has been deemed too large to make this an effective approach. While one can argue that there is a NMEA message type containing velocity, the gathered data has so few occurrences that this evaluation disregards it.

5.1.2 Validity of the Formalised Domain Rules

To test the proposed methods, the possible scenario configurations are segmented to cover all relevant cases.

Simulated Data Sets

First are the domain-dependent segmentations by the formalised Rules 12 a) i), ii), iii); 13 b); 14 a); and 15.

The details of the generated scenarios are:

- vessel count v

  This element represents the number of participating vessels; for the simulated

scenarios, two were chosen to simplify evaluation.

- rules r
  Count of the formalised rules applicable to the scenario.

- configurations c
  The usual configurations include ships that approach each other because they are head on, angled towards each other, or one is faster and headed toward the other, with the exception of overtaking, in which one follows the other. There are two or three possibilities per vessel covered here, depending on the scenario.

- wind directions w
  These are handled in relative terms as seen from the vessels, either port or starboard. There are 2 possibilities (luv or lee) per vessel, which are usually covered by the generation of one scenario with interchangeable vessels.

An agent behaviour r is generated for each of the vessels v, leading to $\|r\| * \|v\|$ agent behaviour descriptions. The different spatial configurations of the ships must be considered for each of the rules. The ships can be head-on towards each other, one can follow the other, or they can be in an angular configuration c, generating $\|r\| * \|v\| * \|c\|$ different scenarios. Additionally, the modality of the wind w for each sailing vessel divides the test space into even more sets. The worst case results in $\|r\| * \|v\| * \|c\| * \|w\|$ scenarios to test. The actual number is less than the total possible amount due to identical combinations, for example, with wind directions; compare with Table 5.2. The angular values are approximate, caused by drawing the test scenarios within the GUI and based on the possible different spatial configurations as mentioned in Section 4.3. This is an intended effect, bringing in a degree of uncertainty similar to real-world scenarios in which angular configurations are also approximate, depending on sensory equipment or its absence. The angular values have been chosen to cover the common situations as expected by the COLREGS.

The provided predicates for the data set contained:

- vessel(VESSEL)
  The identifier for a naval vessel

- heading(TIME, VESSEL, CARDINAL)
  The cardinal information about the orientation of a naval vessel at a time point

- opra(TIME, VESSEL_A, VESSEL_B, ORIENTATION_A_B, ORIENTATION_B_A) for vessels
  The $\mathcal{OPRA}$ relation between two vessels A and B.
- opra(TIME, VESSEL_A, wind, s, ORIENTATION_B_A) for wind direction
  The $\mathcal{OPRA}$ same relation denoting the incoming wind direction for a vessel. The direction of B can be omitted here, as the COLREGS are not concerned with the angle at which the wind is facing the ship and only regard the direction of the wind as seen from the ship.
- closeEnoughToCollide(TIME, VESSEL_A, VESSEL_B)
  This relation denotes that two vessels are sufficiently close to each other that the COLREGS are applicable.

	COLREGS Rule	Formalised Process	Angular Configuration
Rule 12	(a), (i)	$change\_course\_rule12\_i(T, X)$ $keep\_course\_rule12\_i(T, X)$	Tested for 90°, 180°, 270°
	(a), (ii)	$change\_course\_rule12\_ii(T, X)$ $keep\_course\_rule12\_ii(T, X)$	Tested for 45°, 90°
	(a), (iii)	$change\_course\_rule12\_iii(T, X)$	Tested for 90°, 135°, 270°
Rule 13	(b)	$change\_course\_rule13\_b(T, X)$ $keep\_course\_rule13\_b(T, X)$	Tested for 0°, 22°, 338°
Rule 14	(a)	$change\_course\_rule14\_a(T, X)$ $keep\_course\_rule14\_a(T, X)$	Tested for 158°, 180°, 202°
Rule 15	(full)	$change\_course\_rule15(T, X)$	Tested for 45°, 90°, 335°

Table 5.2 – Process Description Coverage. Angular Configurations represent approximated values.

Generation of the Synthetic Data Sets

Scenarios were generated for their ability to validate the functionality of the proposed analytical methods. The primary requirement was the successful detection of spatial configurations relevant to the COLREGS. Also, it was tested whether there were erroneous

detections for rules, and if the behaviour with respect to computational time and space was satisfactory. The scenarios were also used to gather indications for further improvements in the best case. The synthesised scenarios were tested against the previously described abstract agent behaviour descriptions and the outcome checked with respect to found agent behaviour instances (correct hits as well as false positives), generated logic predicates, and runtime. I requested the assistance of a domain expert, namely Podebry (2014) [2] to ensure the correct interpretation of the COLREGS and thereby a robust agent behaviour description base. The scenario-generation procedure is described in detail below.

Trajectory recording

A scenario is recorded by forming multiple trajectories for ships, setting points along the intended line in the evaluation tool. The evaluation tool "Scenario Creator 0.1" is a graphical interface written in Python[3]. It offers recording of the named point-based lines, their conversion into continuous splines by simple cubic spline interpolation, their conversion back into discrete abstracted qualitative facts, and the output into ASP-readable syntax.

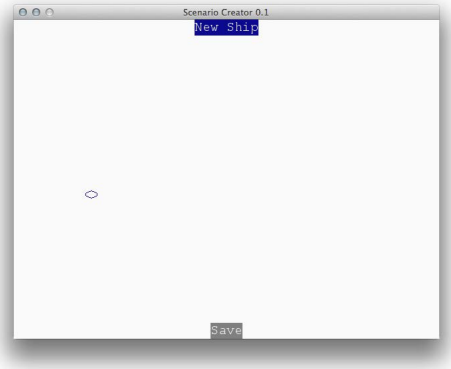


Figure 5.1 – Scenario Creator 0.1 - Empty scenario.

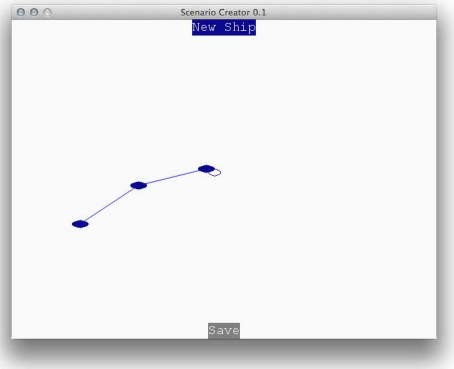


Figure 5.2 – Scenario Creator 0.1 - Starting a trajectory.

A typical generation procedure works a follows: The user first sees the welcome screen of the Scenario Creator 0.1 tool. Indicated in the upper middle is the current ship colour, in this case blue, on the button for insertion of a new ship. The mouse indicator shows an empty oval, hovering over the position for the first way point, Figure 5.1. The

[2] currently active helmsman for AIDA cruises 'club resort' ships, April 22, 2014
[3] Python 2.6.7 and its packages scipy, numpy, pygame, to name only the most important ones

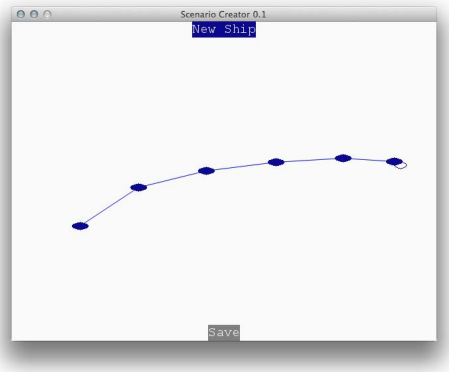


Figure 5.3 – Scenario Creator 0.1 - Finishing a trajectory.

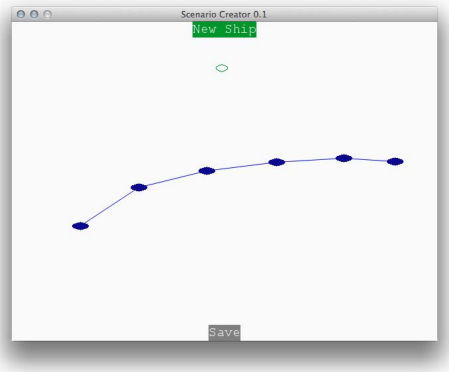


Figure 5.4 – Scenario Creator 0.1 - Switching to the next vessel.

user proceeds by placing the desired number of points into the window, Figure 5.2 - 5.3. When finished with the first vessel, the user clicks the "New Ship" button in the upper middle. When switching to the next vessel, the indicator for the next points and the button change colour to aid in differentiating between lines, Figure 5.2.

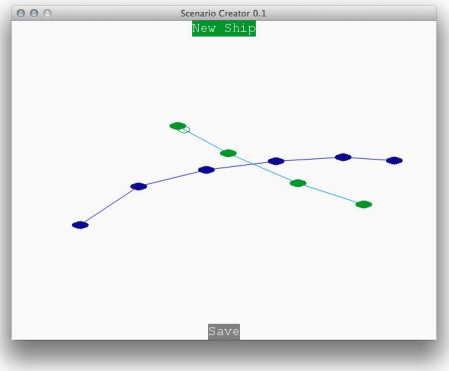


Figure 5.5 – Scenario Creator 0.1 - Intersecting second trajectory.

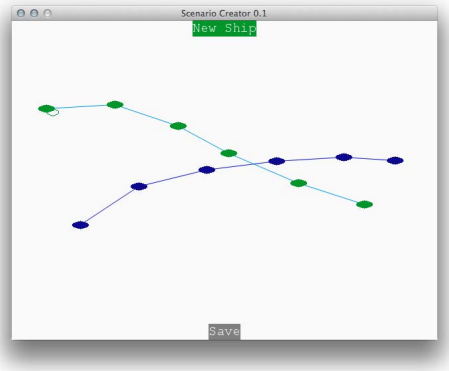


Figure 5.6 – Scenario Creator 0.1 - Finalised scenario.

The insertion of way points for further vessels is done in the same manner. Once all desired trajectories have been entered, the user can start the conversion process, which ends with saving the ASP-conform observation data under an appropriate filename.

A noteworthy point here is that the entered trajectories have an inherent vagueness with respect to the angular and positional configuration of the vessels. The intention is to model the non-ideal situations found in the real world.

Results for the Validation Analysis

To determine the validity of the formalised rules, assessment of a fulfilled condition is based on two elements.

First, the visual component was observed for display of the correct analysis. Second, the log files were evaluated to verify results. Test criteria:

1. Has the correct rule been identified?

2. Were the actors correctly assigned?

3. Were no rules other than the intended one found (false positives)?

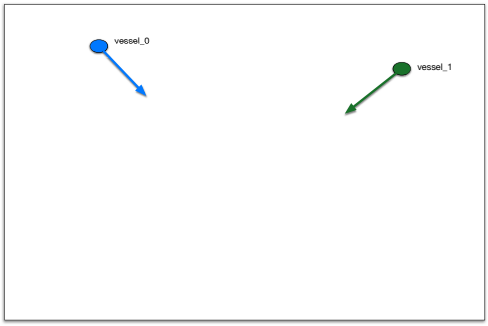


Figure 5.7 – (Visibility revised) screenshot from simulation scenario Rule 12, 90°, timestamp 0, start.

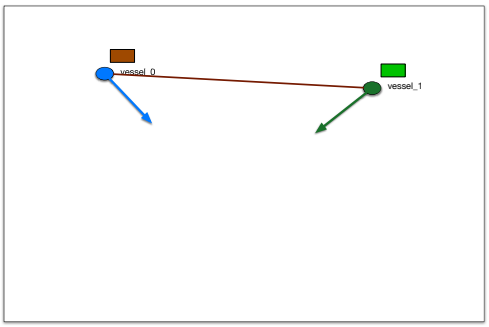


Figure 5.8 – (Visibility revised) screenshot from simulation scenario Rule 12, 90°, timestamp 8, flags for $change$ and $keep$ course, $couldCollide$ predicate active.

A typical result is shown in Figure 5.7 to 5.10. Figure 5.7 shows the start configuration of two ships, in this case, two sailing vessels in an angular configuration of roughly 90° (COLREGS Rule 12 (a) (i)). Figure 5.8 is eight time steps into the future and displays the distance threshold (red line connecting the ships) as well as the firing of Rule 12 (a) (i) itself, as the conditions for an avoidance manoeuvre by vessel 0 and the keep course directive for vessel 1 are fulfilled. This is symbolised by a brown flag for vessel 0, the vessel that must avoid, and a green flag for vessel 1 to indicate that its course should be kept. Figure 5.9 depicts the avoidance movement in progress with active flags and Figure 5.10 shows both vessels back on their respective courses. All rules from Table 5.2 have been evaluated and the results are listed in Table 5.3 .

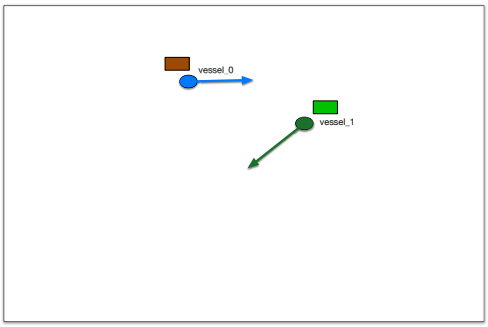


Figure 5.9 – Screenshot from simulation scenario Rule 12, 90°, timestamp 24, flags for *change* and *keep* course active, avoidance behaviour in progress.

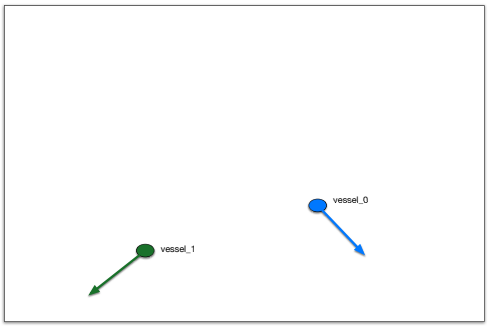


Figure 5.10 – Screenshot from simulation scenario Rule 12, 90°, timestamp 76, vessels back on course.

	COLREGS Rule & Angle	Matched Rule with (time, rule)	Test Cleared
Rule 12	(a), (i), 90°	$ruleFired(8, rule12\_i)$	✓
	(a), (i), 180°	$ruleFired(16, rule12\_i)$	✓
	(a), (i), 270°	$ruleFired(8, rule12\_i)$	✓
	(a), (ii), 45°	$ruleFired(11, rule12\_ii)$	✓
	(a), (ii), 90°	$ruleFired(1, rule12\_ii)$	✓
	(a), (iii), 90°	$ruleFired(21, rule12\_iii)$	✓
	(a), (iii), 135°	$ruleFired(12, rule12\_iii)$	✓
	(a), (iii), 270°	$ruleFired(12, rule12\_iii)$	✓
Rule 13	(b), 0°	$ruleFired(1, rule13\_b)$	✓
	(b), 22°	$ruleFired(3, rule13\_b)$	✓
	(b), 338°	$ruleFired(1, rule13\_b)$	✓
Rule 14	(a), 158°	$ruleFired(18, rule14\_a)$	✓
	(a), 180°	$ruleFired(17, rule14\_a)$	✓
	(a), 202°	$ruleFired(15, rule14\_a)$	✓
Rule 15	45°	$ruleFired(26, rule14\_a)$	□
	90°	$ruleFired(12, rule15)$	✓
	335°	$ruleFired(1, rule15)$	✓

Table 5.3 – Process Description Coverage. Angular degree values represent approximated values. Only the first occurrence of a $ruleFired(time, rule)$ predicate is noted, as ships once engaged are blocked for other rules.

Discussion

The results of the evaluation indicate the functionality of the agent behaviour description recognition for synthetic data. Except for one case, the correct rules were found. This

results in 94.11% positive matching and a 100% coverage of domain rules. The failed test for *Rule 15 with an angle of 45°* was investigated and showed that although the correct preconditions for *Rule 15* were fulfilled and the rule was instantiated, it was suppressed by an earlier firing of *Rule 14 a)*. This turned out to be a granularity issue; choosing a larger m for the $\mathcal{OPRA}_m$ relation, resulting in a larger number of sections and in turn a finer differentiation of spatial configurations, fixed it. The evaluation shown above retains the granularity of $\mathcal{OPRA}_4$ to highlight this issue. For evaluation purposes, the ambiguity of the overlapping interpretation of the rules is an interesting result and underscores the importance of a domain-dependent granularity decision. Bear in mind that the generation of the simulated scenarios was deliberately done without precisely measured angles, to include a certain amount of vagueness. The main goal of this part of the evaluation was to demonstrate the validity of the agent behaviour analysis in a controlled environment. This has been shown successfully.

5.2 Applicability to Real-World Data

To confirm that the qualitative analysis developed in this thesis is able to handle real-world data, the methods were next evaluated with a record of automated inter-ship communication in the form of AIS messages. The use of these messages is mandatory on commercial vessels and widespread on sailing ships. AIS is specifically designed to counter poor visibility conditions through a broadcast of positional data and optional attached data about the vessel. Following my analysis framework, the metric data was abstracted, formalised into predicates, and combined with the formalised rule sets. The goal of this evaluation was to demonstrate the applicability of the qualitative analysis on a real-world scenario.

The system running the tests was a MacBook Pro, 13-inch, mid-2012 model with a 2.9 GHz Intel Core i7, 8 GB 1600 Mhz DDR3 ram, running OS X 10.9.3.

5.2.1 Data Description

This part of the evaluation was made on real-world data, namely recorded National Marine Electronics Association (NMEA) data sets. To ease interpretation, NMEA data sets were reduced to events occurring between the coordinates Longitude 50.5° to 51.8° and Latitude 1° to 2.2°. The data were recorded from the live NMEA data feed from the AIS Hub Data Sharing Center[4] and were scouted for the parts relevant to the qualitative

[4] `www.aishub.net`, last verified October 19, 2015

analysis. The relevant data was extracted, then fed into the same analysis system as the simulated data. In contrast to Section 5.1.2, the data was not used to show the validity of the recognised situations but rather to prove that the introduced methods can handle real-world data and large data sets.

Abstraction of Sensor Data

As discussed in Chapter 4.1, the AIS system relies on several different data packages. These are encoded in the NMEA 0183 standard format[5]. This format has been adopted as the basis for communication between seafaring vessels and as a universal format for communication between GPS receivers and computers, as well. At a minimum, it usually encodes an identifier (MMSI) for a ship, its GPS position, its state (anchored, sailing, constrained by draft, etc.), and several optional features like velocity, heading, cargo, and so on.

Typical NMEA-standard AIS messages received from other vessels look like this:

```
...
!AIVDM,1,1,,A,137HD:000q2GBF‘CmVn0‘@V20>‘<,0*45
!AIVDM,1,1,,A,13HOI??P0000Vt<LCn:dAOwp0>‘<,0*40
!AIVDM,1,1,,B,33LBbpU0001sKW0RV95:gUan0>‘<,0*6F
!AIVDM,1,1,,A,14‘wsl00igWJ2Rp0d8uc8‘l20>‘<,0*06
!AIVDM,1,1,,A,35NOrdU000rA9@0>g=7SA9‘40>‘<,0*56
!AIVDM,1,1,,A,B52M5:@00FC@MaT>P3ea;wP5kP06,0*11
...
```

Taking as an example the first line of the NMEA sample above, the comma-separated elements are:

[5]The NMEA format can be found under `http://www.nmea.org/Assets/011309-0183_manufacturer_codes.pdf`, checked May 15, 2014. Version 0183 is currently the most widespread, introduced after 0180 and 0182 and now being slowly replaced by NMEA 2000.

!AIVDM	The NMEA message type
1	Number of Sentences (some messages need more than one)
1	Sentence Number (1 unless it is a multi-sentence message)
	The blank is the Sequential Message ID (for multi-sentence messages)
A	The AIS Channel (A or B)
1e7HD:00...	The Encoded AIS Data
0*	End of Data
45	NMEA Checksum

The most interesting information here is stored in the encoded AIS data part. Each ASCII character corresponds to a six-digit binary bit number. For 137..., that would make 1 = 000001, e = 101101, 7 = 000111 and so forth. These have to be converted into decimal numbers and segmented according to the specifications. For this dissertation, a Python implementation by the GPSD project[6] was modified and used to retrieve the MMSI identifier, GPS coordinates, true heading, velocity, time, and status.

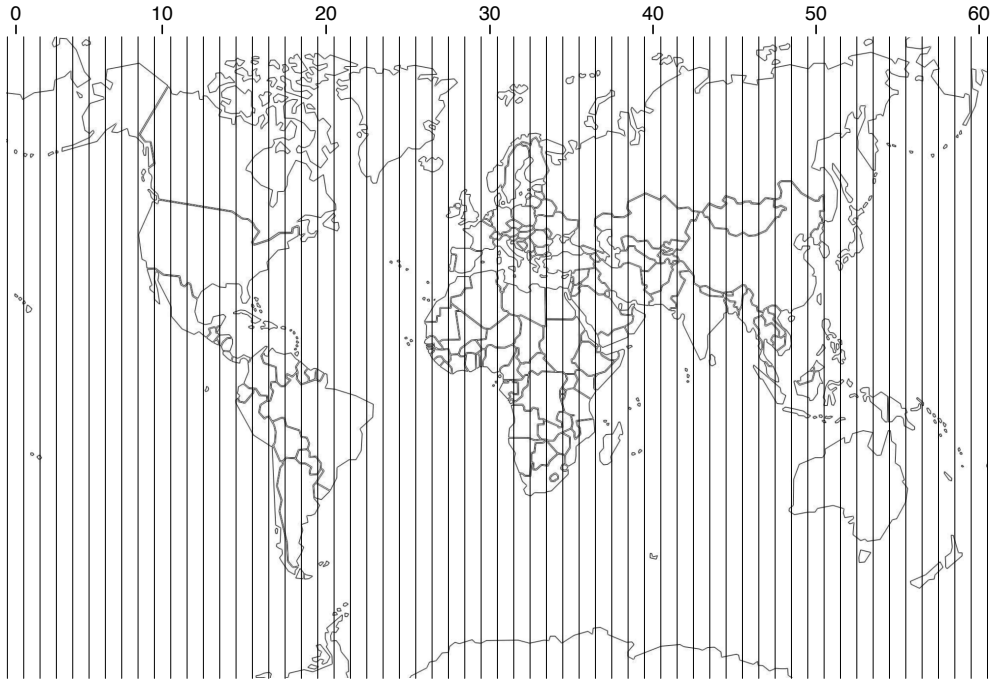


Figure 5.11 – View of the 60 UTM longitude conversion zones.

From here, the GPS coordinates of the associated vessels were converted into the Universal Transverse Mercator (UTM) projected coordinate system. Transfer into a two-

[6] `http://catb.org/gpsd/`, checked May 15, 2014.

dimensional Cartesian coordinate system was necessary to conform to the relative agent behaviour descriptions made by the domain experts, as the rules assume a plane. By applying the conversion into UTM zones, the distortion that occurs when projecting a globe onto a plane is significantly decreased. This is normally without consequence for the reasoning process as the scale of ship interactions is sufficiently small in comparison to $1/60$ of the world's circumference, but the scale is more distorted when approaching the poles and the evaluation is more straightforward without the distortion. The UTM conversion splits the earth into 60 zones, each $6°$ of longitude in width, see Figure 5.11. Note that the schematic was drawn by hand and might differ from the official one. The Earth schematic is already distorted with respect to the true spherical body mapping. The positions were then converted by applying transverse Mercator map projection, a process not discussed in detail here as it is not the focus of this work. The important feature of this projection is that the angular data is preserved while the size of landmarks can be distorted.

In summary, the positions relative to the zones result in a planar two-dimensional projection, that is well suited for the qualitative analysis task. An excerpt is depicted in Figure 5.12, showing ship traffic over three days in the narrow channel between Dover, England, and Calais, France.

Figure 5.12 – Ship traffic example, Dover - Calais

This segment of the map is especially interesting, as the ferries alternating between

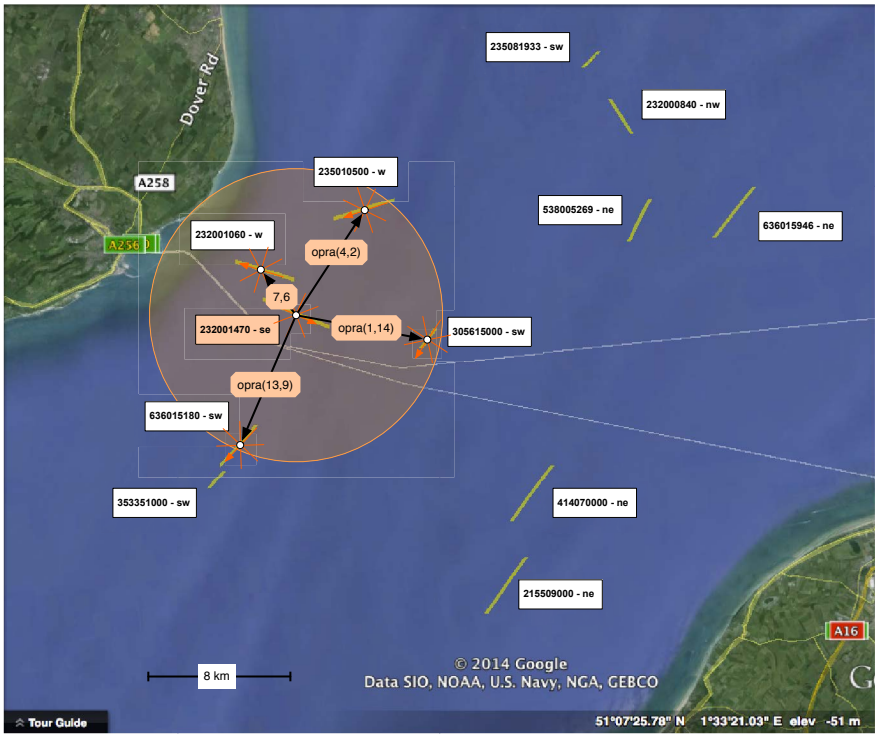


Figure 5.13 – Annotated example, Dover - Calais

Dover and Calais intersect ship traffic from the North Sea and the Atlantic. This creates multiple situations where COLREGS are applicable. The yellow lines symbolise trajectories of individual ships. For reasons of readability, directions of movement are omitted. From here, the generation of the logic primitives is straightforward: relative positional information, a check for the distance threshold for avoidance, and information about the vessel type, if available. The first two were obtained from the GPS coordinates and the latter encoded directly from the AIS data to reduce errors. Additionally, absolute orientation information was retained for each vessel, as it is conveniently encoded in the orientation with respect to metric (angular) cardinal directions.

The relative orientation of the vessels toward each other was calculated by application of trigonometry functions, extracting the angular configuration of two orientated points and projecting it to the $\mathcal{OPRA}$ relation.

The distance was calculated by use of the Haversine Formula (Inman, 1849), based on spherical trigonometry, which hands back the orthodromic distance of the shortest route between two points on the surface of a sphere. This thesis makes the simplification that the earth is round, which is reasonable because of the narrow margin of error due to the

uneven surface of the earth in relation to the threshold distance for collision avoidance. The corresponding logical predicate is $closeEnoughToCollide(time_i, vessel_j, vessel_k)$ and the calculation was done for every pair of ships at every point in time. The vessel type is located in the NMEA data packet or can be deduced from the vessel's status, e.g., 'Under way sailing' is a customary use status that indicates a sailing vessel. If no status was discernible, the vessel's status was set to motorised. Motorised vessels need to avoid all others by the COLREGS, so with respect to safety, that is the soundest interpretation. There are differences for commercial, private and military vessels, but they are outside the focus of this thesis. The relevant logical predicate is $vessel(name_l, type_m)$ with $i, j, k, l, m \in \mathbb{N}$.

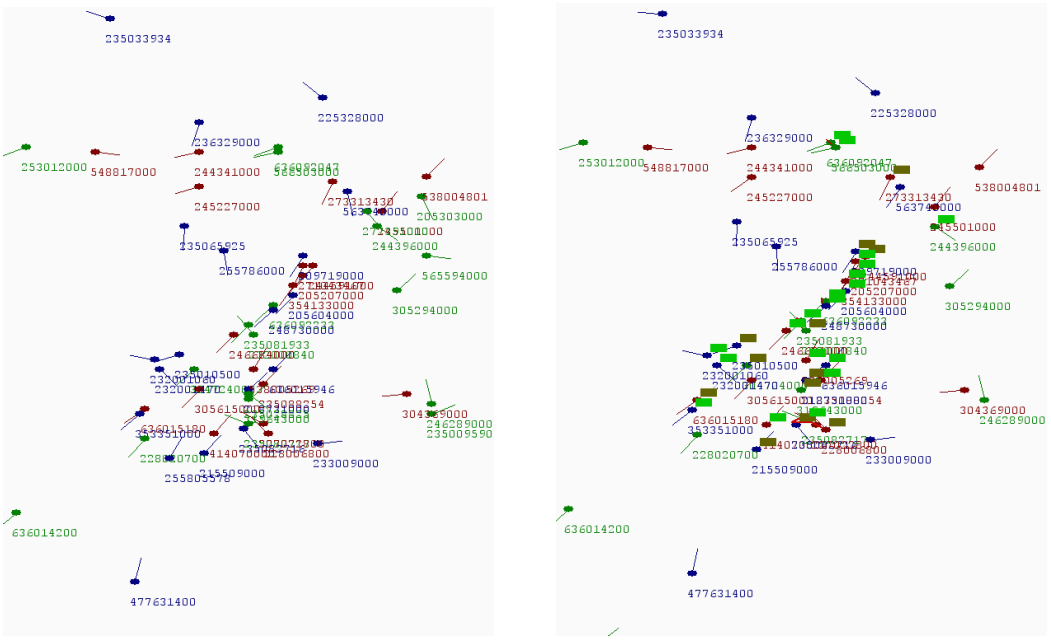
Absolute orientation was converted from the north-orientated degrees in the NMEA packages and qualified into cardinal equivalents like N, NE, E, etc..

Part of the qualified information is depicted in Figure 5.15. NMEA decoded packages were reduced to a time span of one hour for visibility reasons. For the ship with the orange MMSI 232001470 ship identifier and a distance threshold of 8 kilometres (opaque circle), the relevant vessels, their orientation, and the derived $\mathcal{OPRA}$ relations are shown, as derived by the automated qualification implementation. The figure serves as a verified example for the correct transfer from metric measurements to the qualitative representation.

5.2.2 Results of the Analysis

Having applied abstraction to the real-world data, the preconditions for the symbolic agent behaviour analysis are fulfilled. Following the same procedure as for the analysis of the simulated data, the results for ships between the coordinates Longitude 50.5° to 51.8° and Latitude 1° to 2.2° are depicted in the Figures 5.14 and 5.14. The first, Figure 5.14a, shows the ships as blue and red ellipses with their headings (lines extending from the ellipses) and MMSI (ship identifiers). The second, Figure 5.14b, depicts a subset of fired rules: green flags for ships that have to keep their course and brown flags for ships that have to evade.

For an example of the correlation between qualitative representation and agent behaviour detection, Figure 5.15 highlights rule pairs in comparison with the NMEA data visualisation. The extracted predicates have been compared to the AIS data visualisation and found correct. The figure identifies pairs of ships that trigger the corresponding COLREGS rules, indicated by green and brown rectangles. Part of Figure 5.14 is shown enlarged on the right side; the orange explanation boxes contain information taken from



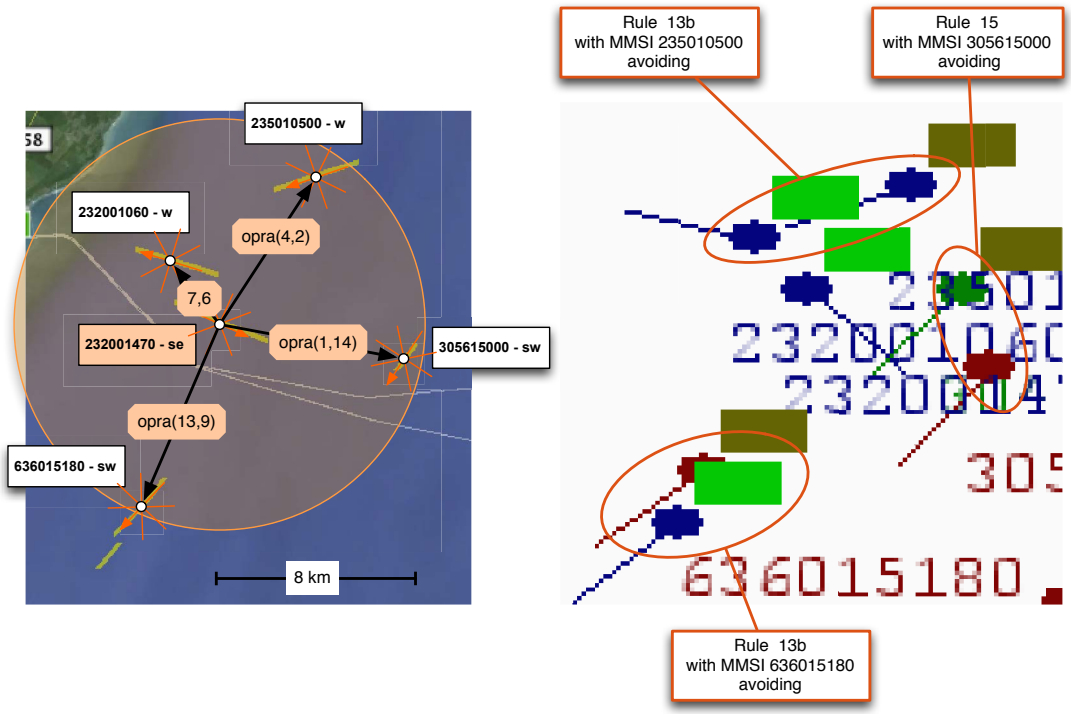
(a) Timestamp 0, ship configurations. **(b)** Timestamp 1, found agent behaviours.

Figure 5.14 – Qualified example visualisation, Dover - Calais

the log files.

5.2.3 Discussion

The evaluation target for the real-world data study was to prove the applicability of the proposed methods for real ship movements. Validity of the abstracted predicates, like $\mathcal{OPRA}_m$ relations, has been shown for all inspected instances in the evaluated scenario. Based on the extracted qualitative data and the previously validated agent behaviour descriptions, process inference has been achieved. However, situations which include movements based on intra-ship communication, for example, cannot be explained due to the fact that such communication is not modelled in the qualitative representation. Therefore, it is reasonable to suppose that not every movement was correctly categorised. That in itself does not present a problem, as the task of this evaluation was to show the applicability of the introduced methods to real-world domain data. The reviewed situations fit the derived facts and the agent behaviour analysis. Given abstracted observations from the domain, the introduced methods worked on the real-world data as well as they did on the simulated scenarios. The main result of this part of the evaluation proves that agent behaviour analysis methods can be applied to real-world data sets.



(a) Hand-drawn annotations of ship arrangements on trajectory visualisation.

(b) Same situation with annotations for visualisation after qualifications and rule test.

Figure 5.15 – Example, Dover - Calais with agent behaviour analysis.

Several examples for the abstraction process can be found in Appendix B.

5.3 Scaling of Data Analysis

This section evaluates the scalability of the agent behaviour analysis implemented in this thesis within the naval domain. The evaluation is done in three steps:

1. Evaluation of computational time, ASP-grounded atoms and rules, as well as the generated predicates from the abstract agent behaviour descriptions for the simulation scenarios.

2. Evaluation using real-world data sets ranging from 10 to 190 vessels, with increases of 10 vessels, based on the same criteria.

3. Evaluation of larger data sets with 50 - 950 vessels, in steps of 50 vessels, again based on the same criteria.

The main focus is the computational time needed to handle the data, as well as the number of generated predicates and grounded atomic ASP facts and rules. The first hypothesis tested here is that the *agent behaviour analysis can be completed in less than 10.0 seconds for a number of ships under 100*. This is relevant to real-world conditions, as the estimated number of ships a steersman has to consider at one time is less than 60 (Podebry, 2014). For larger numbers of ships, the analysis task is evaluated with respect to overall time to completion.

The second hypotheses tests whether the analysis task is able to finish at all for a large number of vessels, with respect to how many ships are usually encountered simultaneously at sea.

5.3.1 Scalability using Simulated Data

Test Description

The test data consists of simulation scenarios as described in section 5.1.2. It is identical to the data set used for the validity evaluation, in which two interacting vessels have 8-12 observations each. The system used for testing was an Intel(R) Xeon(R) CPU X5690 3.47GHz cluster with 24 cores and 142 GB RAM, enabling multiple evaluation runs in parallel. The cluster ran Ubuntu Linux with a 3.2.0-60-generic kernel. The implementation was not tailored for multithreaded execution and ran on single cores each.

Results for Simulated Data Sets

The results of the scaling test for the smallest reasonable scenario configuration are measured in:

Time
The measurement of time representing the complete execution time of the grounder (gringo) and solver (clasp) from the Potassco project Gebser et al. (2011).

Solver Time
This represents the execution time for the solver alone. This shows the difference between the grounding step and the solving itself.

Atoms
An atom[7] A here is in the form of $p(t_1, \ldots, t_n)$, where p is a predicate with an

[7] an atom is any combination of ungrounded, partially, and fully grounded terms.

arity n. It is defined over a language containing predicate and function symbols, constants, and variables as defined in the Potassco manual (Gebser et al., 2008).

Rules
A rule consists of a combination of atoms, connected with first-order logic operators, that is implied on another set of atoms. The convenient notation of Gebser et al. (2008) is: $A_0 \vee \ldots \vee A_l \leftarrow A_{l+1}, \ldots, A_m, not\ A_{m+1}, \ldots, not\ A_n$ with $l, m, n \in \mathbb{N}$.

Crafted Predicates
The count of predicates was crafted for the domain as in Chapter 4 and their base predicates as introduced in Chapter 3.

An overview of the results is displayed in Table 5.4. Notable here is that for the relatively small scenarios based on synthetic data, the solver time was so short that it did not register at all. The column is kept in the table to simplify comparison with the rest of the tables.

Simulated Scenarios

SCENARIO	TIME (SEC)	SOLVER TIME (SEC)	ATOMS	RULES	CRAFTED PREDICATES
rule12-i-180deg	124ms	0,0	15653	15776	2634
rule12-i-270deg	120ms	0,0	16711	17752	2611
rule12-i-90deg	132ms	0,0	16697	17601	2604
rule12-ii-45deg	75ms	0,0	11598	11597	1126
rule12-ii-90deg	111ms	0,0	15715	16220	2677
rule12-iii-135deg	110ms	0,0	13887	15277	1856
rule12-iii-270deg	134ms	0,0	16495	16494	2439
rule12-iii-90deg	120ms	0,0	16097	17133	2130
rule13-c-0deg	91ms	0,0	12806	14457	1446
rule13-c-22deg	98ms	0,0	13926	15972	1699
rule13-c-338deg	105ms	0,0	15026	17179	1856
rule14-a-158deg	93ms	0,0	13906	15310	1763
rule14-a-180deg	87ms	0,0	13887	15125	1760
rule14-a-202deg	100ms	0,0	13915	15352	1781
rule15-135deg	73ms	0,0	12569	12699	1340
rule15-45deg	98ms	0,0	15068	16558	1995
rule15-90deg	64ms	0,0	13626	14110	1532

Table 5.4 – Results of scaling for the simulated data set

The computational time consumed for analysing a given scenario was between 64 ms and 134 ms, compare with Diagram 5.3.1, resulting in a relatively even distribution. So little time was needed to solve the grounded facts that it could not be measured by the internal evaluation mechanisms (interrupts). Atoms and rules both measured in the tenth of thousands of seconds, which is a reasonably small amount of time, congruent with the limited timespan and small number of observations in the simulated scenarios, compare

Diagram 5.3.2. Custom predicates of the domain and their base predicates ranged from 1126 instances for the scenario of Rule 12(ii), up to 2677 instances for the same scenario, see Diagram 5.3.3.

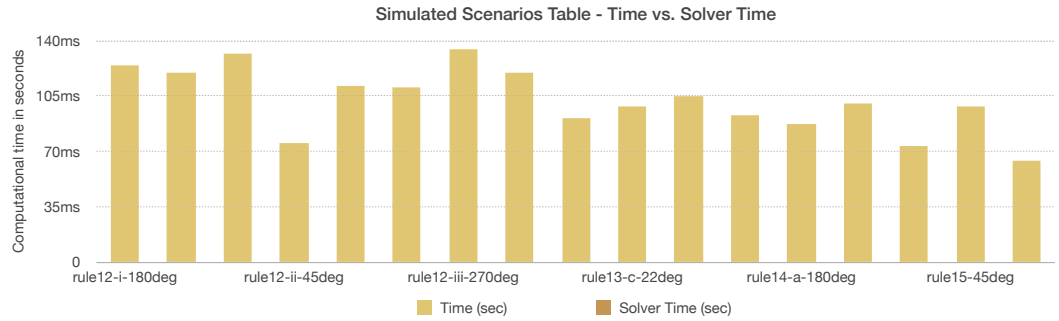


Diagram 5.3.1 – Results of scaling for simulated data set - computation time

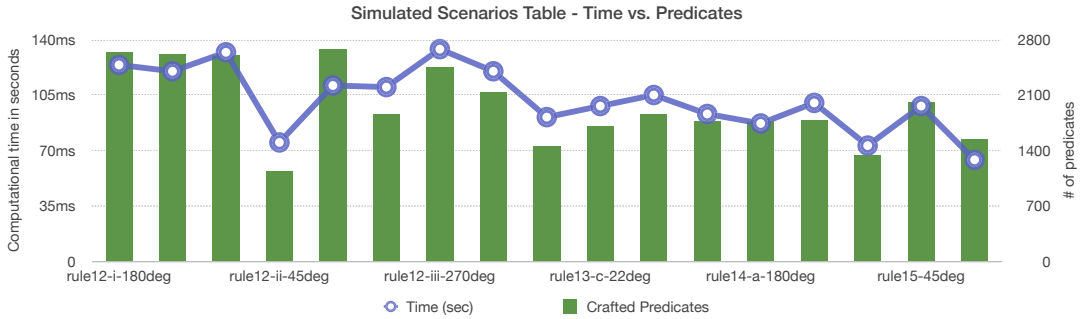


Diagram 5.3.2 – Results of scaling for simulated data set - computation time vs. number of predicates

5.3.2 Scalability using Real Data

Test Description

The test data consists of collections of NMEA messages as described in Section 5.2, but instead of being limited to a small extract between Dover and Calais, they are from all over the Earth. Raw NMEA messages were recorded for 72 hours from March 28 to March 31, 2014, from the AIShub[8] website. Overall, 21,509,865 NMEA messages were available. All duplicate entries had already been filtered out. The average number of packets per

[8] `http://www.aishub.net/`, last checked 14th November 2014.

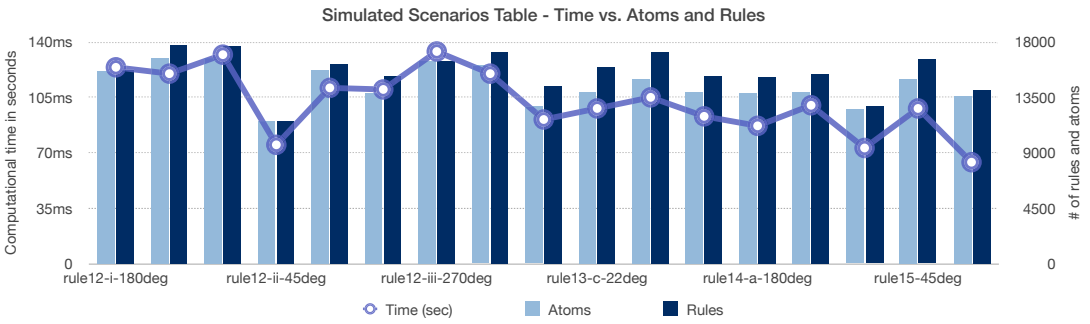


Diagram 5.3.3 – Results of scaling for simulated data set - computation time vs. number of atoms and rules

minute was 4979. The evaluation criteria were identical to those used in the simulation scenarios, Section 5.3.1. Scenarios were evaluated with an increasing number of naval vessels over the same amount of measured time, increasing from 10 vessels up to 190 vessels in steps of 10 vessels. The system setup is the same cluster used in Section 5.3.1.

Results for Real Data Sets

Real Scenarios

VESSELS	TIME (SEC)	SOLVER TIME (SEC)	ATOMS	RULES	PREDICATES
10	820ms	0w	237235	237234	12964
20	2s 127ms	10ms	320776	320775	50729
30	2s 663ms	20ms	356905	356904	76618
40	3s 880ms	30ms	425600	436001	129642
50	3s 832ms	40ms	470464	496079	166729
60	4s 509ms	60ms	545073	628282	228787
70	4s 846ms	70ms	596430	699095	271966
80	5s 755ms	90ms	704001	806666	362558
90	5s 886ms	100ms	750694	853779	401445
100	7s 432ms	120ms	832851	935936	473666
110	7s 107ms	120ms	889378	997151	523686
120	9s 387ms	150ms	1023404	1208740	641500
130	10s 38ms	170ms	1093105	1278910	705241
140	9s 438ms	190ms	1199785	1388685	802800
150	10s 970ms	230ms	1373058	1574614	958325
160	12s 400ms	270ms	1512820	1715040	1086632
170	13s 788ms	290ms	1616759	1818979	1181718
180	13s 382ms	310ms	1772036	2025953	1324863
190	15s 617ms	380ms	2020163	2340018	1552974

Table 5.5 – Scaling for real data sets

As expected, larger scenarios required longer computation times for the detailed

analysis of a scenario. I evaluated a growth factor between single size steps with the equation 5.1, defining growth as the percent difference between two size steps.

$$\text{growth factor}_x = \frac{f_{x+1} - f_x}{f_x} \tag{5.1}$$

The time growth factor between test runs has an arithmetic mean of 21,35 % over all scenarios, with no indication of increasing over the evaluated scenarios. This includes scenarios that have a slightly larger number of vessels with a slightly smaller execution time, which is attributed to the highly efficient problem-solving mechanics of the ASP implementation. The execution time ranged from 0.82 seconds up to 15.617 seconds for the largest scenario with 190 vessels. The solver time mimics the features of the overall computation time.

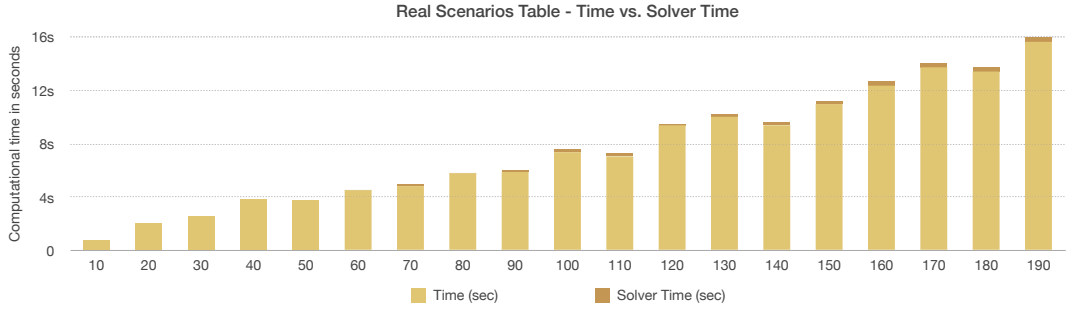


Diagram 5.3.4 – Scaling for real data set - computation time

Growth factors for atoms and rules are slightly smaller, with 11.50 % and 12.55 %, respectively; see Diagram 5.3.7. Compared to the relative growth of the number of vessels, this is noteworthy. As the number of vessels increases by 10 per scenario, thus a declining growth factor, computational time, atom, and rule sets grow with a similar declining factor, see Diagram 5.3.7. The evaluation of computational time shows the time growth in Diagram 5.3.4. Number of predicates, as well as the atoms and rules versus time diagrams, Diagram 5.3.5 and 5.3.6, indicate similar behaviour.

5.3.3 Scalability using Large Real Data Sets

Test Description

The test data setup is the same as that of the previous test, however the scaling is done in steps of 50 vessels up to a total of 950 vessels, as depicted in Table 5.6.

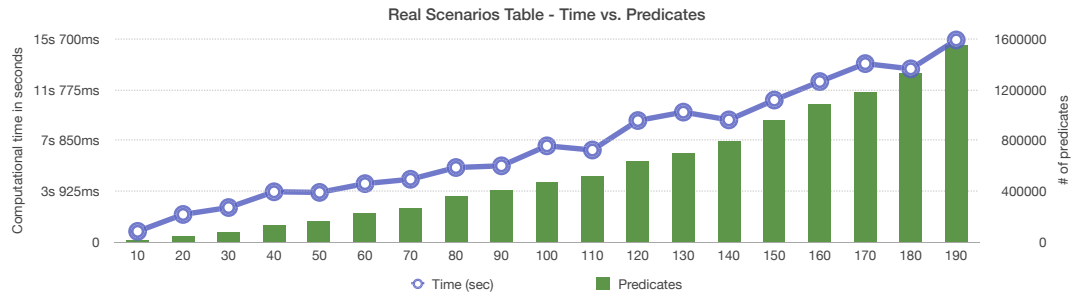


Diagram 5.3.5 – Scaling for real data set - computation time vs. number of predicates

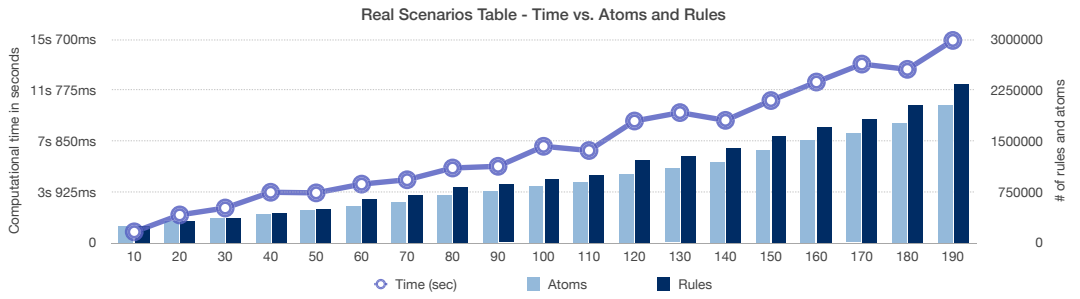


Diagram 5.3.6 – Scaling for real data set - computation time vs. number of atoms and rules

Results for Large Real Data Sets

An overview of the results is shown in Table 5.6. Again, the main criteria for evaluation is the computational time needed for execution of the analysis task. As depicted in Figure 5.3.8, the change in computational time needed for the analysis increases relatively evenly with a declining factor, as seen in Figure 5.3.11. This is to be expected, as the relative increase in vessels says the same with a growing pool of vessels present. Thus, the change gets smaller. Apart from the scenarios with 600 vessels, the increase in time ranges from 33% moving to 150 vessels to only 9% moving to 950 vessels. Execution time ranges from 7.43 seconds for 100 vessels to 3 minutes, 21.76 seconds for 950 vessels. Time for the solver mimics the overall time.

The increase of predicates, as well as atoms and rules created, was similar to that of computational time. The number of atoms increased with scenario size, ranging from 832.851 atoms for 100 vessels to 36,261,131 for 950 vessels. Rules scaled from 935,936 to 55,023,589.

The number of predicates showed an increase as well, ranging from 473,666 to 34,526,506.

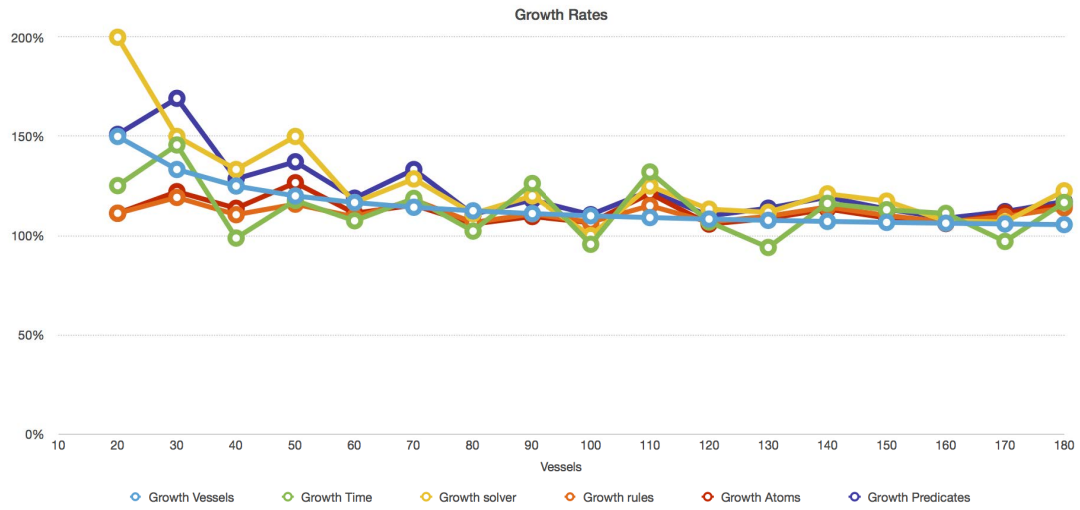


Diagram 5.3.7 – Scaling for real data set - growth factors

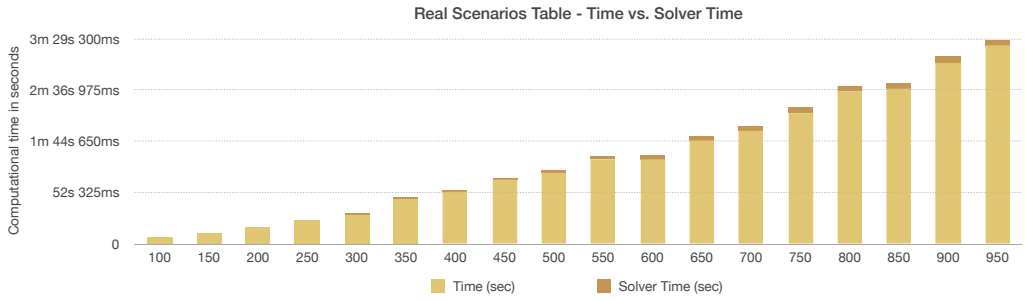


Diagram 5.3.8 – Scaling for real data large set - computation time

The growth spike in the number of atoms, rules, and predicates for the scenario with 300 vessels (Figure 5.3.11) can be attributed to the density of ships in comparison to the scenario with 250 vessels, resulting in a stronger increase in relations between them. These features appear to level out over larger data sets.

5.3.4 Discussion

The scaling performance of the introduced methods is satisfactory with respect to the increase in resource use with number of ships. An identical increase in the number of ships at each step of the scaling resulted in a declining growth factor for execution time and number of atoms, rules, and predicates. This indicates a very well-behaved scaling in terms of computational performance, especially considering that the number of ships that

Real Scenarios Large

VESSELS	TIME (SEC)	SOLVER TIME (SEC)	ATOMS	RULES	PREDICATES
100	7s 432ms	120ms	832851	935936	473666
150	10s 970ms	230ms	1373058	1574614	958325
200	16s 186ms	450ms	2278408	2638002	1792393
250	22s 995ms	650ms	3248787	3846095	2693694
300	29s 248ms	920ms	4621593	5722037	3984014
350	46s 143ms	1s 350ms	6249736	8143840	5521236
400	52s 680ms	1s 690ms	7869963	10397912	7060200
450	1m 4s 841ms	2s 370ms	10237204	14322333	9316575
500	1m 12s 278ms	2s 440ms	11548477	16209551	10581550
550	1m 26s 93ms	2s 780ms	12631015	17576431	11629222
600	1m 26s 372ms	3s 120ms	14695201	20654278	13627096
650	1m 46s 4ms	3s 660ms	17178679	23930830	16027640
700	1m 54s 615ms	3s 970ms	19766725	28857135	18515571
750	2m 13s 703ms	4s 810ms	23538660	34183393	22165827
800	2m 34s 989ms	5s 460ms	26574705	39772328	25100826
850	2m 37s 994ms	6s 90ms	29987361	44194368	28427452
900	3m 4s 59ms	6s 860ms	33409200	50358148	31741231
950	3m 21s 756ms	7s 490ms	36261131	55023589	34526506

Table 5.6 – Results of scaling for large real data sets

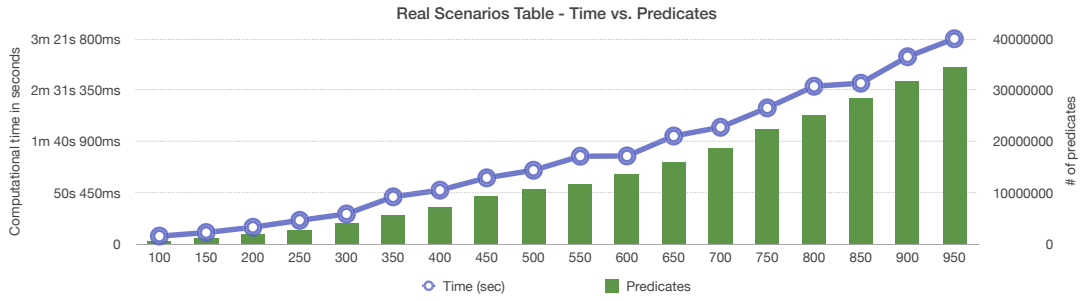


Diagram 5.3.9 – Scaling for large real data set - computation time vs. number of predicates

must to be considered simultaneously in field systems is usually less than 60 vessels.

5.4 Chapter Summary

This chapter methodically proved that the proposed formalisation covers all relevant parts of the COLREGS domain rules. After describing possible outcomes for the rule coverage of the available agent behaviours in the domain, the actual coverage of the formalised agent behaviour descriptions was evaluated. From the full set of available behaviour descriptions, only a subpart was found to be relevant with respect to spatiotemporal concerns. Those relevant parts, concerning interactions between ships, have been fully

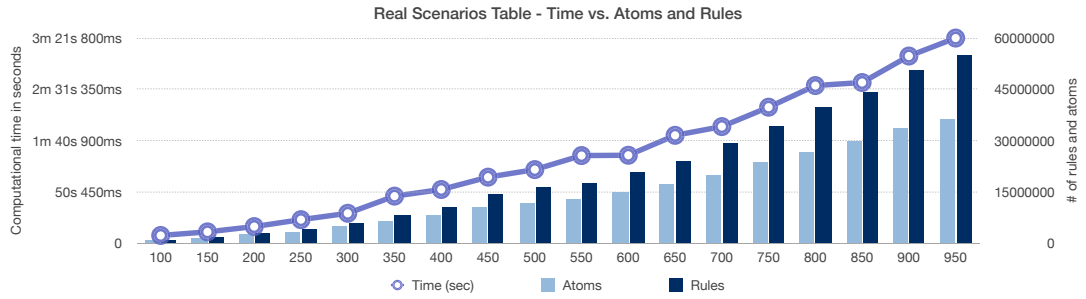


Diagram 5.3.10 – Scaling for large real data set - computation time vs. number of atoms and rules

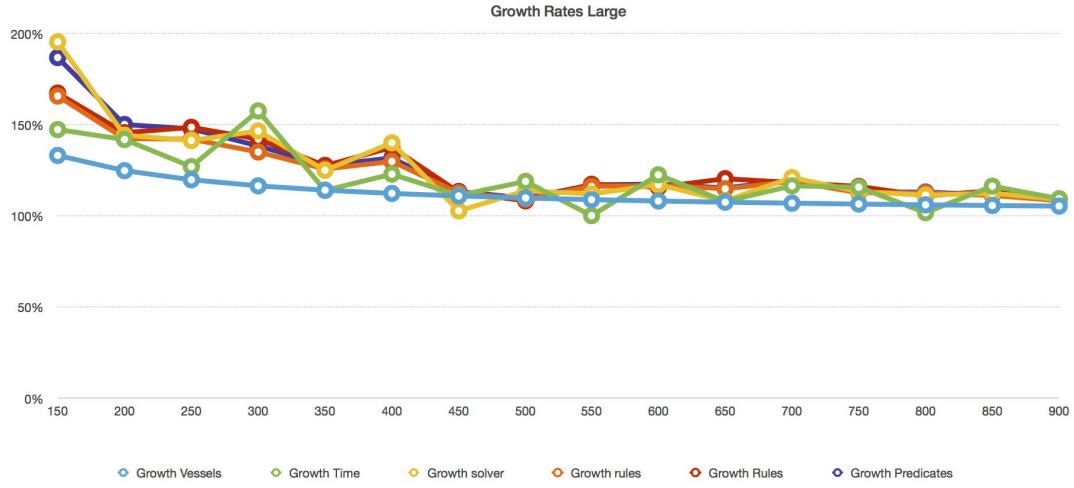


Diagram 5.3.11 – Scaling for large real data set - growth factors for the large evaluation

formalised.

Formalised agent behaviour descriptions were tested in controlled, synthetically generated scenarios to evaluate the suitability of the formalised rules. After reaching only 94.11 % correct matchings, further investigation revealed that the correct rule would have been instantiated (the formalised requirements were correct and met), but it was suppressed by another rule. The abstraction generated configurations that were suitable for both rules, thus constructing an ambiguous situation. A finer granularity for the spatial representation based on $\mathcal{OPRA}$ relations solved the ambiguity, but was not further explored after the solution was found. No unexplainable scenarios were found.

Furthermore, it was shown that the analysis framework not only works on synthetic data, but also on abstracted real-world observations from the naval domain. The findings

are twofold:

1. the abstraction component was able to correctly generate the appropriate qualitative representation to model the scenario.

2. the analysis process could integrate the qualitative observations with the formalised agent behaviour descriptions, resulting in instantiated agent behaviours that handed the analyst high-level descriptions of which processes had occurred in the observed domain.

This can be evaluated further by utilising the analysis component of the framework. An extensive analysis for correct behaviour was not done due to external constraints (inter-ship communication was not recorded) and the fact that a naval domain specialist would have been needed. However, the main goal of showing functionality in a real-world domain has been achieved.

Finally, an extensive scaling evaluation was done, including up to 950 ships moving simultaneously over a three-day period, distributed over the whole globe. While losing the near-real-time analysis (>4s 509ms) possible in standard navigation scenarios (up to 60 ships in the immediate vicinity), the execution time was below >3m 21s 756ms, thus enabling evaluation of large-scale data sets.

Chapter 6

Conclusion

This chapter summarises the contributions of this thesis to the field of qualitative analysis of dynamic domains and evaluates application of the novel analysis method in the naval domain. Furthermore, it contains an outlook for future research, including deployment of the introduced methods in other domains and important considerations when converting the methods into a more general approach.

6.1 Contributions

The ability of agents, human or artificial, to asses their situation with respect to their environment is vital to the performance of goal-directed behaviour, especially in dynamic real-world domains. This thesis presents a robust qualitative approach to analyse spatiotemporal scenarios in a dynamic domain. To this end, spatiotemporal primitives have been introduced, encapsulating well-known methods for representation and reasoning. These predicates are used to generate high-level agent behaviour descriptions which formalise known processes that occur in the domain. By combining formal agent behaviour descriptions with abstracted (agent) observations from the domain, actual instances of the described behaviours can be inferred. This provides several benefits:

- **High-level analysis for data-rich scenarios.**
  Results of the analysis consists of instances of found agent behaviour, thus providing a description of agent dynamics.

- **Robust analysis based on abstracted data.**
  The abstraction of sensor measurements limits information clutter, i.e., acts as Oc-

cam's razor (Thorburn, 1918), removing unneeded information from the inference process. Furthermore, robustness is supported by the design principle of matching observations to descriptions, thus ignoring inherently unmatchable outliers and making use of fitting data if present.

- **Transparent results.**
  By decomposing found agent behaviour descriptions into their fundamental building blocks, the elements leading to the result are listed, detailing how a result came to be.

- **Efficient reasoning.**
  Analyses are based on efficient model-checking algorithms and are thus able to process large-scale data scenarios[1].

With respect to the research questions stated in the Introduction, the following answers have been found:

How can information, available in the domain itself, be represented for automatic qualitative analysis of dynamic scenarios, robust against noisy and sparse sensor data?

The foundation for automated analysis was the development of predicates to encapsulate the information available in the naval domain (COLREGS), generating high-level formal behaviour descriptions. The applicability of the introduced predicates has been demonstrated for vessel movements in accordance with the COLREGS. Thus, for the first part of the comprehensive case study, naval rules defined using natural language by domain experts, i.e., the COLREGS, have been formalised into process descriptions via the encapsulation predicates and their atomic parts.

How can analysis of dynamic scenarios be automated, such that the result contains high-level explanations of processes that occur around an agent?

In the second part of the case study, the formalised rules were extensively tested. After testing the coverage of the domain rules, the validity of the formalised process descriptions was evaluated using simulated scenarios of vessel movements. The generation of simulated scenarios was supervised by a domain expert to ensure correctness. The

[1] With respect to the presented domain.

formalised rules achieved 94.11% positive matchings with 100% coverage of relevant domain rules, see section 5.1.2.

Next, the system was tested against real-world data sets. The data sets consisted of vessel movements recorded worldwide by the AIS collision-avoidance system over a three-day period. Including detailed descriptions of the abstraction process from metric NMEA messages to qualitative facts, the second part of the evaluation provides evidence that the qualitative process analysis is applicable to real-world data sets.

Further evaluation demonstrated the scalability of the analysis framework. The implementation performed its analysis of the trajectories of 100 vessels over three days in less than 10 seconds on a standard desktop computer, see Section 5.2. This result demonstrates that this framework could be utilised as a support system for decision making; a domain expert indicated that a helmsman usually handles no more than 60 - 70 ships in parallel and that 10 seconds is an acceptable timeframe for decision making while underway. When analysing even larger data sets of nearly 1000 ships in parallel, the system still handled grounding of facts and model checking in less than 4 minutes. Matching instances in the grounded data were found in a significantly shorter time, from 2.10% to 3.85% of the grounding time with a tendency to take proportionally longer with larger data sets.

What formal modelling language is needed to represent behaviour in the context of a dynamic scenario more intuitively than that using pure numerical models, and how can observations be combined with rule descriptions to enable high-level semantic domain analysis?

The most noteworthy feature of this work is the generality of the analysis method. Specifics of the domain are handled within the process descriptions and in the abstraction of metric measurements to qualitative facts. The analysis process itself is domain independent. The inference method for finding process instances, on which the analysis is based, can be applied to any domain that provides formalisable descriptions for processes and sensor data that can be abstracted to match the predicates. By combining rule formalisation and abstracted observations, a more intuitive modelling of agent behaviours as well as a high-level analysis have been achieved.

In summary, my contributions to the main challenges posed in this thesis are:

- Based on the linear temporal logic syntax and the well-researched $\mathcal{OPRA}_m$ calculus

as a representation for orientated points, I developed predicates for the intuitive construction of formal process descriptions in a dynamic domain. These predicates form the basis for a qualitative analysis system, improving understanding of dynamic environments due to the following factors:

  - The introduced predicates ease process formalisation for non-experts in the field of logic. This is achieved by the use of already present semantic rules, avoiding interoperation with numerical data.
  - The qualitative analysis provides evaluation of processes, the actors involved, and the time frames when the processes occurred, thus greatly enhancing the situational awareness of an observing agent, as well as facilitating later analysis of recorded scenarios.
  - Two methods, reasoning and proximity-based insertion, generate hypothetical facts that enrich the knowledge base of observations needed for the process analysis, indicate missing observations, or both.
  - Based on model-checking algorithms, a robust and efficient inference system has been introduced to find process instances in potentially large and incomplete metric data sets.

- A comprehensive case study in the naval domain demonstrates the functionality of the approach and the deployed methods. The system was evaluated from three perspectives:

  - Domain coverage and validity using synthetic data.
  - Applicability to real-world data.
  - System behaviour when confronted with large (900+ ships, 3 days) data sets.

6.2 Future Work

This work opens new avenues for further research. Although it is generally more intuitive for humans to understand natural language than large number sets, it would be worthwhile to evaluate in depth how well the introduced primitives ease the generation of process descriptions for different domains. To this end, a study would need to evaluate different approaches to model a domain in order to analyse the processes that occur. Ideally, one would present several probabilistic approaches like Bayesian Nets, symbolic approaches like QTC representations, and the methods from this thesis to domain experts

from different professions. Adaptability and ease of modelling could be empirically measured, and the results would give insight into the benefits and drawbacks of the different methods in various domains.

Another branch for further research is to combine process description generation with a statistical evaluation to generate the most likely explanations in the case of ambitious agent behaviour descriptions. An example of this is the hospital domain, in which visitor movement data sets can be obtained but in-hospital process descriptions are lacking.

The hospital domain also provides an interesting scenario for semi-automatic simulation generation to populate a simulated environment with agents that behave according to validated behaviour patterns. An example application is the flow simulation suite created and used by the Frankfurt Airport to identify bottlenecks in the architecture with regard to passenger flows.

Another direction for research is evaluation of how grounded sets from model-checking algorithms could be reused to minimise execution time. The overwhelming majority of computation time for the analysis is used to generate grounded facts, and the actual search for instantiated agent behaviour descriptions is small in comparison. If there were a way to limit generation of grounded facts to change the delta of two similar scenarios, overall execution time could drop significantly. This would be advantageous if the analysis system is to be integrated into real-time support systems on ship bridges.

Finally, data integration and results representation offer additional research opportunities. For example, the integration of linked data sets (Kauppinen and de Espindola, 2011) for meta-information can provide formally sound metadata for the predicates, such as bridging the semantical gap between different types of motor boats or definitions for spatial arrangements. By integrating linked data knowledge with the formalisation process, modelling can be made less ambiguous and inference reasoning enhanced for the generalisation of observed data elements. As there exist different visualisation techniques for linked data sets, an interesting direction for further development would be to convert found results into linked-data-set-conforming information. By achieving this, use of those visual representations would become feasible.

A handy addition to the introduced framework would be knowledge about coastal features or static elements such as landmasses, islands, lighthouses, and the like. The current state of this work only considers open, unobstructed sea. Integration could be achieved by accessing topological maps and comparing ship positions with the spatial information from the maps. Utilising this additional knowledge, more trajectories could be explained. This would require an enhanced set of formalised agent behaviour descriptions as well, because ships have a different rule set when using waterways.

Bibliography

Alejo, D., Cobano, J. A., Heredia, G. and Ollero, A. (2013). Particle Swarm Optimization for collision-free 4D trajectory planning in Unmanned Aerial Vehicles, *International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 298–307.

Allen, J. F. (1983). Maintaining knowledge about temporal intervals, *Communications of the ACM* **26**(11): 832–843.

Allen, J. F. and Hayes, P. J. (1985). A Common-Sense Theory of Time., *International Joint Conferences on Artificial Intelligence* pp. 528–539.

Antoniotti, M. and Mishra, B. (1995). Discrete event models + temporal logic = supervisory controller: Automatic synthesis of locomotion controllers, *Proceedings of the IEEE Conference on Robotics and Automation (ICRA)*, pp. 1441–1446.

Balcan, M.-F. and Blum, A. (2010). A Discriminative Model for Semi-Supervised Learning, *Journal of the ACM (JACM)* **57**(3): 1–46.

Bennewitz, M., Burgard, W., Cielniak, G. and Thrun, S. (2005). Learning Motion Patterns of People for Compliant Robot Motion, *The International Journal of Robotics Research (IJRR)* **24**(31): 39–41.

Berg, J. P. v. d. and Zijm, W. H. M. (1999). Models for warehouse management: Classification and examples, *International Journal of Production Economics* **59**(1-3): 519–528.

Bhatt, M. and Dylla, F. (2009). A Qualitative Model of Dynamic Scene Analysis and Interpretation in Ambient Intelligence Systems, *International Journal of Robotics and Automation: Special Issue on Robotics for Ambient Intelligence* **24**(3).

Bredeweg, B. and Struss, P. (2004). Current topics in qualitative reasoning, *AI Magazine* **24**(4): 13–16.

Chen, J., Cohn, A., Liu, D., Wang, S., Ouyang, J. and Qiiangyuan, Y. (2015). A survey of qualitative spatial representations, *The Knowledge Engineering Review* **1**(01): 106–136.

Cimatti, A., Giunchiglia, E., Giunchiglia, F. and Traverso, P. (1997). Planning via model checking: A decision procedure for AR, *Recent Advances in AI planning*, Springer Verlag, Berlin/Heidelberg, pp. 130–142.

Clarke, E. M., Emerson, E. A. and Sistla, A. P. (1986). Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications, *ACM Transactions on Programming Languages and Systems* **8**(2): 244–263.

Clementini, E., Di Felice, P. and Hernández, D. (1997). Qualitative representation of positional information, *Artificial Intelligence* **95**(2): 317–356.

Colonius, I. (2009). *Diploma Thesis -Lokalisierung auf Basis kontinuierlicher und diskreter Drehwinkelveränderungen zwischen Landmarken in semistrukturierten Umgebungen*, University of Bremen, Fachbereich 3, Mathematics and Computer Science.

Colonius, I. (2012). Hypotheses Generation for Process Recognition in a Domain Specified by Temporal Logic, *in* L. Frommberger, K. Schill and B. Scholz-Reiter (eds), *AILog Workshop at ECAI 2012*, Montpellier, France, pp. 49–54.

Dasgupta, P., Chakrabarti, P. P. and Deka, J. K. (2002). Min-max event-triggered computation tree logic, *Sadhana-Academy Proceedings in Engineering Sciences* **27**: 163–180.

Dubba, K. S. R., Bhatt, M., Dylla, F., Cohn, A. and Hogg, D. (2011). Interleaved Inductive-Abductive Reasoning for Learning Event-Based Activity Models, *21st International Conference on Inductive Logic Programming (ILP 2011)* .

Dubois, D. and Prade, H. (2001). Possibility theory, probability theory and multiple-valued logics: a clarification, *Annals of Mathematics and Artificial Intelligence* **32**(1-4): 35–66.

Dylla, F. (2008). *An Agent Control Perspective on Qualitative Spatial Reasoning*, Vol. 320 of *DISKI*, Akademische Verlagsgesellschaft Aka GmbH (IOS Press), Heidelberg, Germany.

Edelkamp, S. and Jabbar, S. (2006). Action Planning for Directed Model Checking of Petri Nets, *Electronic Notes in Theoretical Computer Science* **149**(2): 3–18.

Fagin, R., Halpern, J. Y. and Megiddo, N. (1988). A logic for reasoning about probabilities, *Logic in Computer Science, 1988. LICS '88., Proceedings of the Third Annual Symposium on*, pp. 410–421.

Forbus, K. D. (1996). *Qualitative Reasoning*, The Computer Science and Engineering Handbook, CRC Press.

Frausto, J., Elizalde, F. and Reyes, A. (2008). A Logic Formal Validation Model for the Explanations Generation in an Intelligent Assistant, *International Conference on Artificial Intelligence*, pp. 9–14.

Freksa, C. (1992a). Temporal reasoning based on semi-intervals, *Artificial Intelligence* **54**(1-2): 199–227.

Freksa, C. (1992b). Using Orientation Information for Qualitative Spatial Reasoning, *in* A. U. Frank, I. Campari and U. Formentini (eds), *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 162–178.

Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T. and Thiele, S. (2008). A user's guide to gringo, clasp, clingo, and iclingo, *Technical report*.

Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T. and Schneider, M. (2011). Potassco: The Potsdam Answer Set Solving Collection, *AI Communications* **24**(2): 107–124.

Gelfond, M. and Kahl, Y. (2014). *Knowledge representation, reasoning, and the design of intelligent agents: the answer-set programming approach*, Cambridge University Press, New York, NY.

Glez-Cabrera, F. J., Álvarez-Bravo, J. V. and Díaz, F. (2013). QRPC: A new qualitative model for representing motion patterns, *Expert systems with applications* **40**(11): 4547–4561.

Goldszmidt, M. and Pearl, J. (1996). Qualitative probabilities for default reasoning, belief revision, and causal modeling, *Artificial Intelligence* **84**: 57–112.

Gottfried, B. (2004). Reasoning about intervals in two dimensions, *International Conference on Systems, Man and Cybernetics, IEEE*, pp. 5324–5332.

Halle, S., Ngoupe, E. L., Villemaire, R. and Cherkaoui, O. (2013). Distributed firewall anomaly detection through LTL model checking, *International Symposium on Integrated Network Management (IM), IFIP/IEEE* pp. 194–201.

Harati-Mokhtari, A., Wall, A., Brooks, P. and Wang, J. (2007). Automatic Identification System (AIS): Data Reliability and Human Error Implications, *The Journal of Navigation* **60**(03): 373.

Hasegawa, K., Hata, K., Niwa, K. and Fukuto, J. (2008). Transmission evaluation of Ship-borne Automatic Identification System (AIS) in congested waterways, *2008 8th International Conference on ITS Telecommunications (ITST)*, IEEE, pp. 18–23.

Hirsh, H. (1987). Explanation-based generalization in a logic-programming environment., *Proceedings of the 10th international joint conference on Artificial intelligence* **1**: 221–227.

Ilic-Stepic, A. (2010). A logic for reasoning about qualitative probability, *Publications de l'Institut Mathematique* **1**(87): 97–108.

Inman, J. (1849). *Navigation and Nautical Astronomy*, For the Use of British Seamen, 7th edn, Francis and John Rivington, London.

International Maritime Organization (IMO) (2003). *COLREG : Convention on the International Regulations for Preventing Collisions at Sea, 1972*, 4th edn, International Maritime Organization (IMO).

Kauppinen, T. and de Espindola, G. M. (2011). Linked Open Science-Communicating, Sharing and Evaluating Data, Methods and Results for Executable Papers, *Proceedings of the International Conference on Computational Science, ICCS* .

Kim, J., Lee, Y. and Moon, I. (2008). Automatic Verification of Biochemical Network Using Model Checking Method, *Chinese Journal of Chemical Engineering* **16**(1): 90–94.

Kloetzer, M. and Belta, C. (2006). LTL planning for groups of robots, *Proceedings of the IEEE International Conference on Networking, Sensing and Control (ICNSC)*, pp. 578–583.

Kloetzer, M. and Belta, C. (2010). Automatic Deployment of Distributed Teams of Robots From Temporal Logic Motion Specifications, *IEEE Transactions on Robotics* **26**(1): 48–61.

Kress-Gazit, H. and Wongpiromsarn, T. (2011). Correct, reactive robot control from abstraction and temporal logic specifications, *Special Issue of the IEEE Robotics and Automation Magazine on Formal Methods for Robotics and Automation* **18**: 65–74.

Kreutzmann, A., Colonius, I., Frommberger, L., Dylla, F., Freksa, C. and Wolter, D. (2011). On Process Recognition by Logical Inference, *European Conference on Mobile Robots*, SFB/TR 8 Spatial Cognition, University of Bremen, Germany, pp. 7–12.

Kreutzmann, A., Colonius, I., Wolter, D., Dylla, F., Frommberger, L. and Freksa, C. (2013). Temporal logic for process specification and recognition, *Intelligent Service Robotics* **6**(1): 5–18.

Kreutzmann, A., Wolter, D., Dylla, F. and Lee, J. H. (2013). Towards Safe Navigation by Formalizing Navigation Rules, *International Journal on Marine Navigation and Safety of Sea Transportation* **7**(2): 161–168.

Kuipers, B. (2000). The Spatial Semantic Hierarchy, *Artificial Intelligence* **119**(1-2): 191–233.

Kuipers, B. and Byun, Y.-T. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations, *Robotics and Autonomous Systems* **8**(1-2): 47–63.

Lahijanian, M., Andersson, S. B. and Belta, C. (2011). Control of Markov decision processes from PCTL specifications, *American Control Conference (ACC), 2011*, pp. 311–316.

LaValle, S. M. (2006). *Planning Algorithms*, Cambridge University Press.

Liao, L., Patterson, D. J., Fox, D. and Kautz, H. (2007). Learning and inferring transportation routines, *Artificial Intelligence* **171**(5–6): 311–331.

Lifschitz, V. (2002). Answer set programming and plan generation, *Artificial Intelligence* **138**(1-2): 39–54.

Lifschitz, V. (2008). What Is Answer Set Programming?, *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence* pp. 1594–1597.

Ligozat, G. F. (1993). Qualitative triangulation for spatial reasoning, *in* A. U. Frank and I. Campari (eds), *Spatial Information Theory. A Theoretical Basis for GIS*, Springer, Berlin, Heidelberg, pp. 54–68.

Marr, D. and Nishihara, H. K. (1978). Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes, *Proceedings of the Royal Society B: Biological Sciences* **200**(1140): 269–294.

Mastrogiovanni, F., Sgorbissa, A. and Zaccaria, R. (2009). Context assessment strategies for Ubiquitous Robots, *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 2717–2722.

Merchant, N. D., Witt, M. J., Blondel, P., Godley, B. J. and Smith, G. H. (2012). Assessing sound exposure from shipping in coastal waters using a single hydrophone and Automatic Identification System (AIS) data, *Marine Pollution Bulletin* **64**(7): 1320–1329.

Moratz, R., Renz, J. and Wolter, D. (2000). Qualitative Spatial Reasoning about Line Segments, *European Conference on Artifical Intelligence* **14**: 39–58.

Mossakowski, T. and Moratz, R. (2012). Qualitative Reasoning about Relative Direction of Oriented Points, *Artificial Intelligence* **180–181**: 34–45.

Nute, D. (1989). Defeasible logic and temporal projection, *Decision Support and Knowledge Based Systems Track, Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences*, Hawaii, pp. 575–581.

Omer, J. and Farges, J.-L. (2013). Hybridization of Nonlinear and Mixed-Integer Linear Programming for Aircraft Separation With Trajectory Recovery, *IEEE Transactions on Intelligent Transportation Systems* **14**(3): 1218–1230.

Pietrzykowski, Z. and Uriasz, J. (2009). Knowledge representation in a ship's navigational decision support system, *Marine Navigation and Safety of Sea Transportation* pp. 45–50.

Pnueli, A. (1977). The temporal logic of programs, *18th Annual Symposium on Foundations of Computer Science* pp. 46–57.

Podebry, C. (2014). Personal Communication.

Rajagopalan, R. and Kuipers, B. (1994). Qualitative spatial reasoning about objects in motion: application to physics problem solving, *Proceedings of the Tenth Conference on Artificial Intelligence for Applications*, Multiple values selected, pp. 238–245.

Randell, D. A., Cui, Z. and Cohn, A. (1992). A Spatial Logic Based on Regions and Connection, *in* B. Nebel, C. Rich and W. Swartout (eds), *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92)*, Morgan Kaufmann, San Mateo, CA, pp. 165–176.

Reiter, R. (1980). A logic for default reasoning, *Artificial Intelligence* **13**(1-2): 81–132.

Richards, A. and How, J. (2002). Aircraft trajectory planning with collision avoidance using mixed integer linear programming, *Proceedings of the American Control Conference, 2002,* pp. 1936–1941.

Rouwenhorst, B., Reuter, B., Stockrahm, V., Van Houtum, G. J., Mantel, R. J. and Zijm, W. H. M. (2000). Warehouse design and control: Framework and literature review, *European Journal of Operational Research* **122**(3): 515–533.

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach (2nd Edition)*, Prentice Hall Series in Artificial Intelligence, Prentice Hall.

Santos, P. E. and Shanahan, M. (2002). Hypothesising object relations from image transitions, *15th European Conference on Artificial Intelligence (ECAI)* pp. 292–296.

Schlieder, C. (1991). *Anordnung und Sichtbarkeit - eine Charakterisierung unvollständigen räumlichen Wissens*, PhD thesis, Computer Science Department, University of Hamburg, Germany.

Schlieder, C. (1993). Representing visible locations for qualitative navigation, *in* N. P. Carrete and M. G. Singh (eds), *Qualitative Reasoning and Decision Technologies*, CIMNE, Barcelona, Barcelona, Spain, pp. 523–532.

Schlieder, C. (1995). Reasoning About Ordering, *in* A. U. Frank and W. Kuhn (eds), *Proceedings of the Conference on Spatial Information Theory (COSIT'95)*, Springer, Berlin, Heidelberg, pp. 341–349.

Smierzchalski, R. and Michalewicz, Z. (2000). Modeling of ship trajectory in collision situations by an evolutionary algorithm., *IEEE Transactions on Evolutionary Computation* **4**(3): 227–241.

Smith, S. L., Tumova, J., Belta, C. and Rus, D. (2010). Optimal path planning under temporal logic constraints, *International Conference on Intelligent Robots and Systems (IROS)* pp. 3288–3293.

Sridhar, M., Cohn, A. and Hogg, D. C. (2011). From Video to RCC8: Exploiting a Distance Based Semantics to Stabilise the Interpretation of Mereotopological Relations, *in* M. Egenhofer (ed.), *Spatial Information Theory*, Springer, Berlin, Heidelberg, pp. 110–125.

Szlapczynski, R. and Szlapczynska, J. (2012). On evolutionary computing in multi-ship trajectory planning., *Applied Intelligence* **37**(2): 155–174.

Tachmazidis, I., Antoniou, G., Flouris, G. and Kotoulas, S. (2012). Towards Parallel Non-monotonic Reasoning with Billions of Facts, *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning* pp. 1–5.

Thorburn, W. M. (1918). The Myth of Occam's razor, *Mind* **27**: 345–353.

Uriasz, J. (2008). Navigational knowledge base, *Scientific Journals of the Maritime University of Szczecin* **13**(85): 92–98.

Van de Weghe, N. (2004). *Representing and Reasoning about Moving Objects: A Qualitative Approach*, PhD thesis, Ghent University.

Van de Weghe, N., Cohn, A. and Maeyer, P. (2005). Representing moving objects in computer-based expert systems: the overtake example, *Expert systems with applications* **29**: 977–983.

Van de Weghe, N., Kuijpers, B., Bogaert, P. and Maeyer, P. (2005). A qualitative trajectory calculus and the composition of its relations, *GeoSpatial Semantics, Lecture Notes in Computer Science* **3799**: 60–76.

Wagner, T. (2006). *Qualitative Sicht-basierte Navigation in unstrukturierten Umgebungen*, PhD thesis, University of Bremen, Faculty 3, Mathematics and Computer Science.

Wallgrün, J. O., Frommberger, L., Wolter, D., Dylla, F. and Freksa, C. (2007). Qualitative Spatial Representation and Reasoning in the SparQ-Toolbox, *Spatial Cognition V: Reasoning, Action, Interaction: International Conference Spatial Cognition 2006*, Bremen, Germany, pp. 1–20.

Yang, Q. (2009). Activity recognition: linking low-level sensors to high-level intelligence, *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence* pp. 20–25.

Appendices

Appendix A

Convention on the International Regulations for Preventing Collisions at Sea, 1972 (COLREGs)

The following is an abbreviated version of the COLREGs to provide details not listed in the work and support the reader in developing an overall picture of the Rules. It is taken directly from the International Maritime Organization (IMO) (2003). Adoption: 20 October 1972; Entry into force: 15 July 1977

The 1972 Convention was designed to update and replace the Collision Regulations of 1960, which were adopted at the same time as the 1960 SOLAS Convention. One of the most important innovations in the 1972 COLREGs was the recognition given to traffic separation schemes. To that end, Rule 10 gives guidance in determining safe speed, the risk of collision, and the conduct of vessels operating in or near traffic separation schemes. The first such traffic separation scheme was established in the Dover Strait in 1967. It was initially operated on a voluntary basis, but in 1971 the IMO Assembly adopted a resolution stating that observance of all traffic separation schemes should be made mandatory. The COLREGs make this obligation clear.

Technical provisions

The COLREGs include 38 rules divided into five sections: Part A - General; Part B - Steering and Sailing; Part C - Lights and Shapes; Part D - Sound and Light signals; and Part E - Exemptions. There are also four Annexes containing technical requirements for

lights and shapes and their positioning, sound signalling appliances, additional signals for fishing vessels when operating in close proximity, and international distress signals.

Part A - General (Rules 1-3)

Rule 1 states that the rules apply to all vessels upon the high seas and all waters connected to the high seas and navigable by seagoing vessels.

Rule 2 covers the responsibility of the master, owner, and crew to comply with the rules.

Rule 3 includes definitions.

Part B- Steering and Sailing (Rules 4-19)

Section 1 - Conduct of vessels in any condition of visibility (Rules 4-10)

Rule 4 says that the section applies in any condition of visibility.

Rule 5 requires that "every vessel shall at all times maintain a proper look-out by sight and hearing as well as by all available means appropriate in the prevailing circumstances and conditions so as to make a full appraisal of the situation and of the risk of collision".

Rule 6 addresses safe speed. It requires that: "Every vessel shall at all times proceed at a safe speed...". The Rule describes the factors which should be taken into account in determining safe speed. Several of these refer specifically to vessels equipped with radar.The importance of using "all available means" is further stressed in

Rule 7 covering risk of collision, which warns that "assumptions shall not be made on the basis of scanty information, especially scanty radar information".

Rule 8 covers actions to be taken to avoid collision.

Rule 9 a vessel proceeding along the course of a narrow channel or fairway is obliged to keep "as near to the outer limit of the channel or fairway which lies on her starboard side as is safe and practicable". The same Rule obliges a vessel of less than 20 metres in length or a sailing vessel not to impede the passage of a vessel "which can safely navigate only within a narrow channel or fairway".

The Rule also forbids ships to cross a narrow channel or fairway "if such crossing impedes the passage of a vessel which can safely navigate only within such channel

or fairway". The meaning of "not to impede" was clarified by an amendment to Rule 8 in 1987. A new paragraph (f) was added, stressing that a vessel which was required to not impede the passage of another vessel should take early action to allow sufficient sea room for the safe passage of the other vessel. Such a vessel must also fulfil this obligation when taking avoidance action in accordance with the steering and sailing rules when risk of collision exists.

Rule 10 of the Collision Regulations deals with the behaviour of vessels in or near traffic separation schemes adopted by the IMO. By Regulation 8 of Chapter V (Safety of Navigation) of SOLAS, the IMO is recognised as being the only organisation competent to manage international measures concerning the routing of ships. The effectiveness of traffic separation schemes can be judged from a study made by the International Association of Institutes of Navigation (IAIN) in 1981. Between 1956 and 1960 there were 60 collisions in the Strait of Dover; twenty years later, following the introduction of traffic separation schemes, the total was only 16.

In other areas where such schemes did not exist, the number of collisions rose sharply. New traffic separation schemes are introduced regularly and existing ones are amended when necessary to respond to changed traffic conditions. To allow this to be done as quickly as possible, the MSC has been authorised to adopt and amend traffic separation schemes on behalf of the IMO.

Rule 10 states that ships crossing traffic lanes are required to do so "as nearly as practicable at right angles to the general direction of traffic flow." This reduces confusion to other ships as to the crossing vessel's intentions and course, and at the same time enables that vessel to cross the lane as quickly as possible.

Fishing vessels "shall not impede the passage of any vessel following a traffic lane" but are not banned from fishing. This is in line with Rule 9, which states that "a vessel engaged in fishing shall not impede the passage of any other vessel navigating within a narrow channel or fairway". In 1981, the regulations were amended. Two new paragraphs were added to Rule 10 to exempt vessels which are restricted in their ability to manoeuvre "when engaged in an operation for the safety of navigation in a traffic separation scheme" or when engaged in cable laying.

In 1987 the regulations were again amended. It was stressed that Rule 10 applies to traffic separation schemes adopted by the IMO and does not relieve any vessel of her obligation under any other rule. It also specified that if a vessel is obliged to cross traffic lanes it should do so as nearly as practicable at right angles to the

general direction of the traffic flow. In 1989 Regulation 10 was further amended to clarify the vessels which may use the "inshore traffic zone."

Section II - Conduct of vessels in sight of one another (Rules 11-18)

Rule 11 says that the section applies to vessels in sight of one another.

Rule 12 states actions to be taken when two sailing vessels are approaching one another.

Rule 13 covers overtaking, specifically that the overtaking vessel should keep out of the way of the vessel being overtaken.

Rule 14-16 Rule 14 deals with head-on situations. Crossing situations are covered by Rule 15 and the action to be taken by the give-way vessel is laid down in Rule 16.

Rule 17 deals with the action of the stand-on vessel, including the provision that the stand-on vessel may "take action to avoid collision by her manoeuvre alone as soon as it becomes apparent to her that the vessel required to keep out of the way is not taking appropriate action".

Rule 18 deals with responsibilities between vessels and includes requirements for vessels which shall keep out of the way of others.

Section III - conduct of vessels in restricted visibility (Rule 19)

Rule 19 states that every vessel should proceed at a safe speed adapted to prevailing circumstances and restricted visibility. A vessel detecting another vessel by radar should determine if there is risk of collision and if so take avoiding action. A vessel hearing the fog signal of another vessel should reduce speed to a minimum.

Part C Lights and Shapes (Rules 20-31)

Rule 20 states that the rules concerning lights apply from sunset to sunrise. [Rule 21] gives definitions.

Rule 22 covers visibility of lights, indicating that lights should be visible at minimum ranges (in nautical miles) determined according to the type of vessel.

Rule 23 covers lights to be carried by power-driven vessels underway.

Rule 24 covers lights for vessels towing and pushing.

Rule 25 covers light requirements for sailing vessels underway and vessels under oars.

Rule 26 covers light requirements for fishing vessels.

Rule 27 covers light requirements for vessels not under command or restricted in their ability to manoeuvre.

Rule 28 covers light requirements for vessels constrained by their draught.

Rule 29 covers light requirements for pilot vessels.

Rule 30 covers light requirements for vessels anchored and aground. [Rule 31] covers light requirements for seaplanes.

Part D - Sound and Light Signals (Rules 32-37)

Rule 32 gives definitions of whistle, short blast, and prolonged blast.

Rule 33 says that vessels 12 metres or more in length should carry a whistle and a bell, and that vessels 100 metres or more in length should, in addition, carry a gong.

Rule 34 covers manoeuvring and warning signals using whistle or lights.

Rule 35 covers sound signals to be used in restricted visibility.

Rule 36 covers signals to be used to attract attention.

Rule 37 covers distress signals.

Part E - Exemptions (Rule 38)

Rule 38 says that ships which comply with the 1960 Collision Regulations and were built or already under construction when the 1972 Collision Regulations entered into force may be exempted from some requirements for light and sound signals for specified periods.

Appendix B

Naval Vessel Movements Extracted from NMEA Data Sets

This section contains visualisations of naval vessel trajectories in Google Earth ©Google, based on the recorded AIS data used in this thesis. Yellow lines are interpolated movements. The figures serve as examples of movements typical in the recorded samples. Note that special movements, such as circular movements by moored vessels, are not inserted into the inference process as long as the ships broadcasted their state.

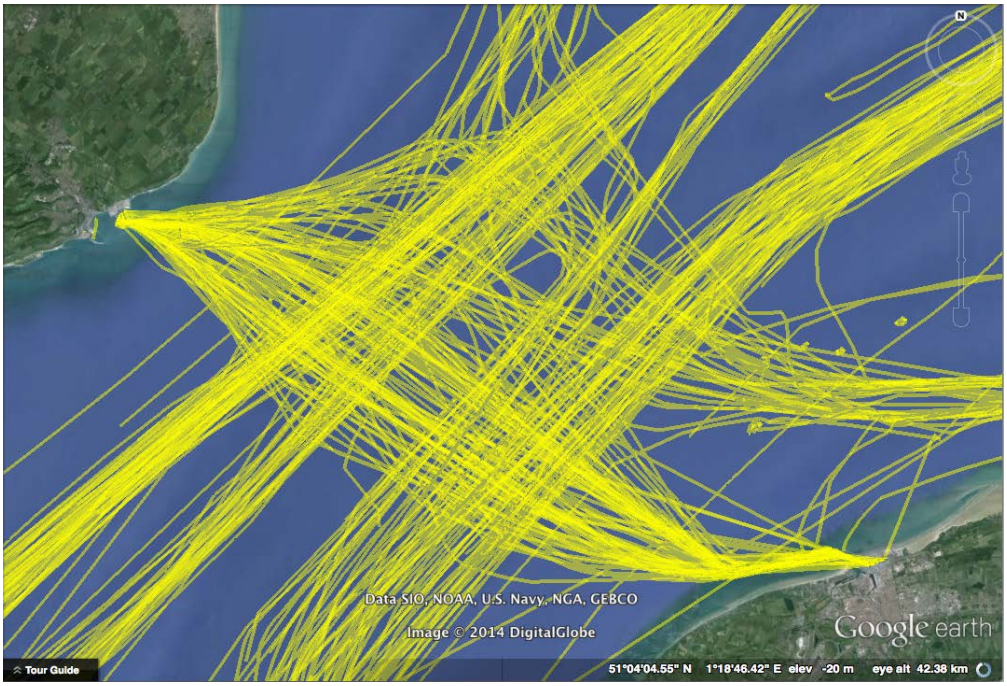


Figure B.1 – Dover - Calais overview

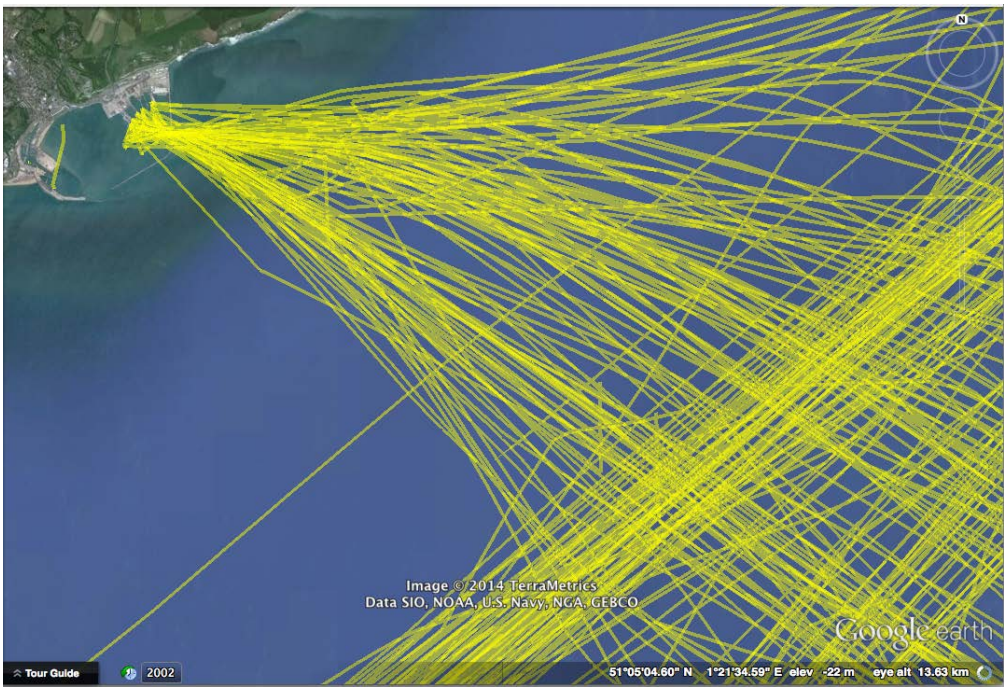


Figure B.2 – Dover.

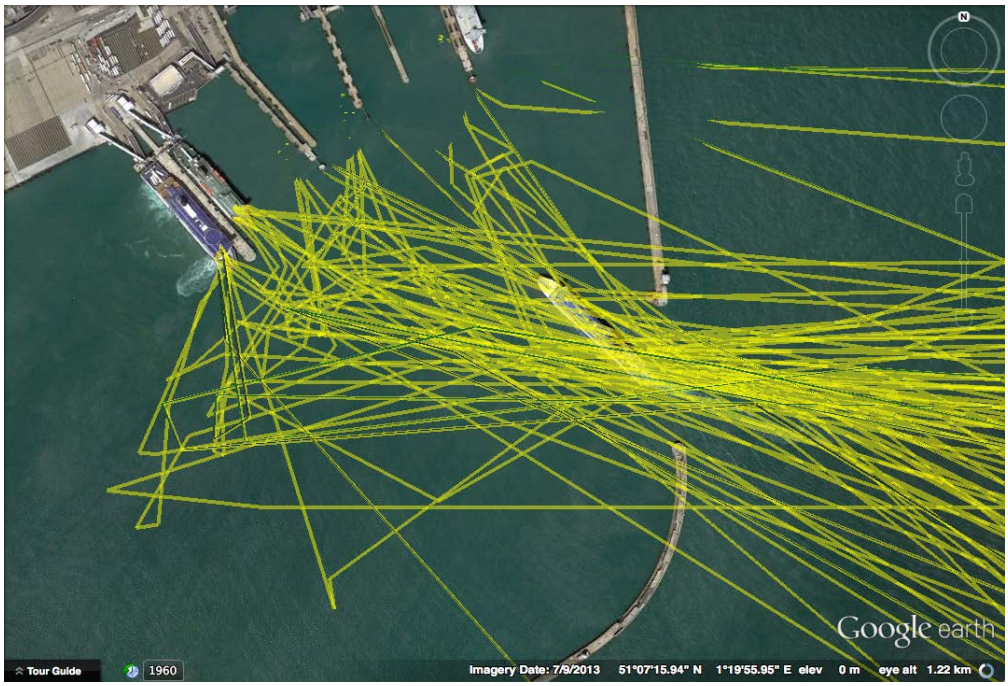


Figure B.3 – Dover harbour - detectable ferry movements at docking point.

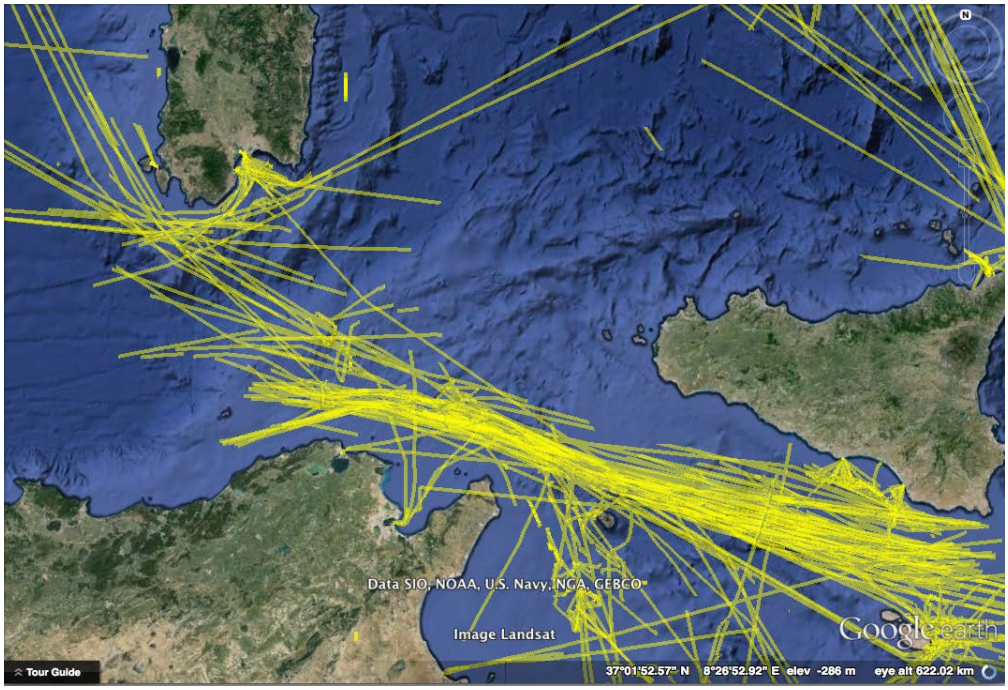


Figure B.4 – Sea beneath Italy.

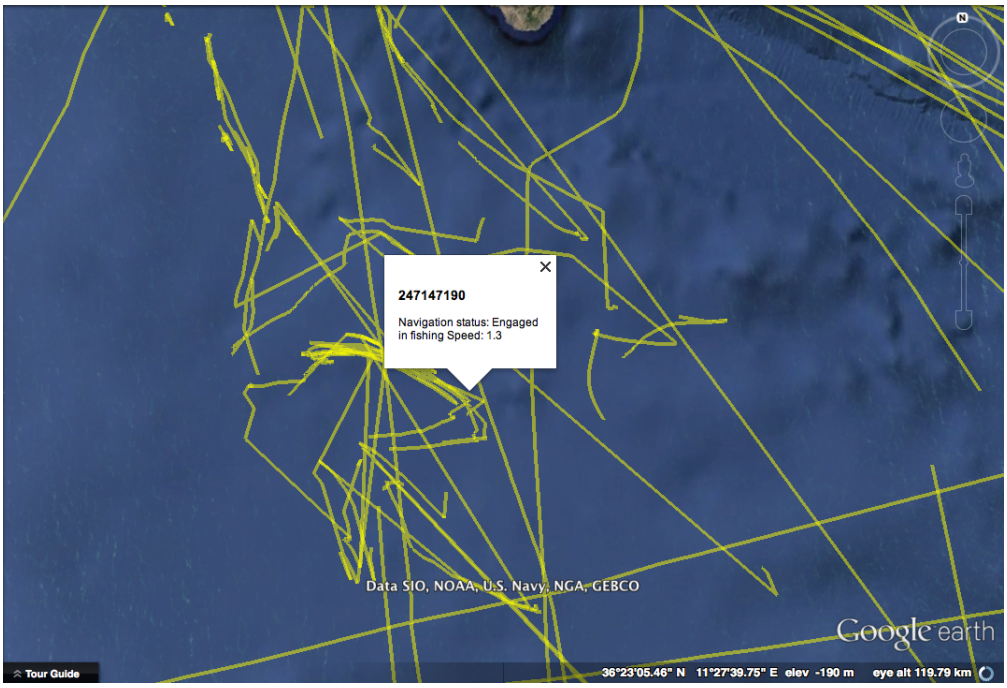


Figure B.5 – Sea beneath Italy - fishing boats.

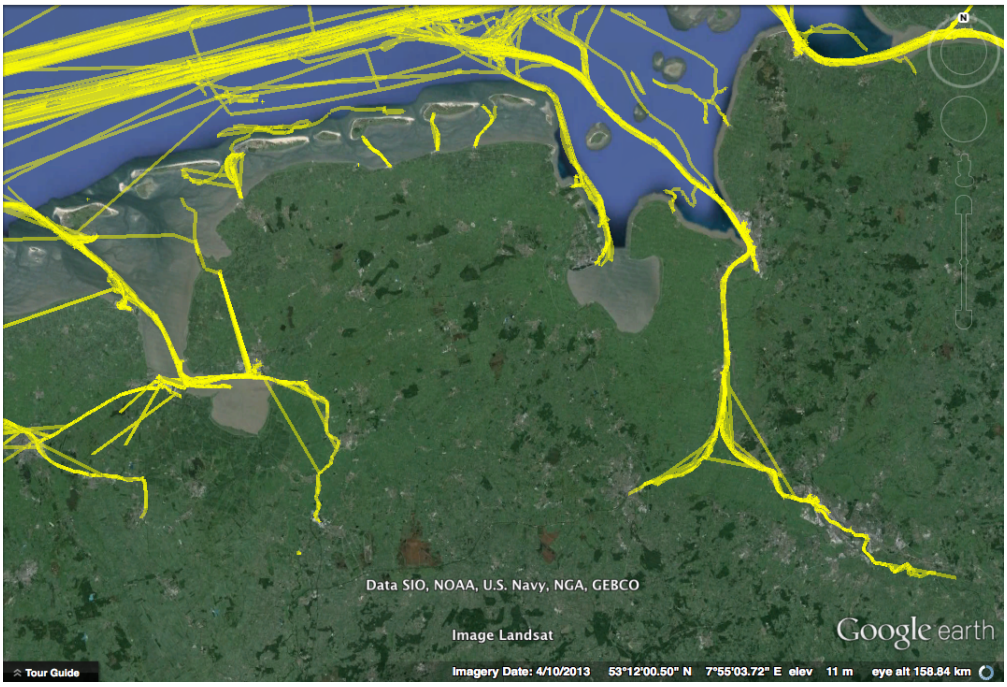


Figure B.6 – Northwest Germany - small lines between coastal islands are ferries.

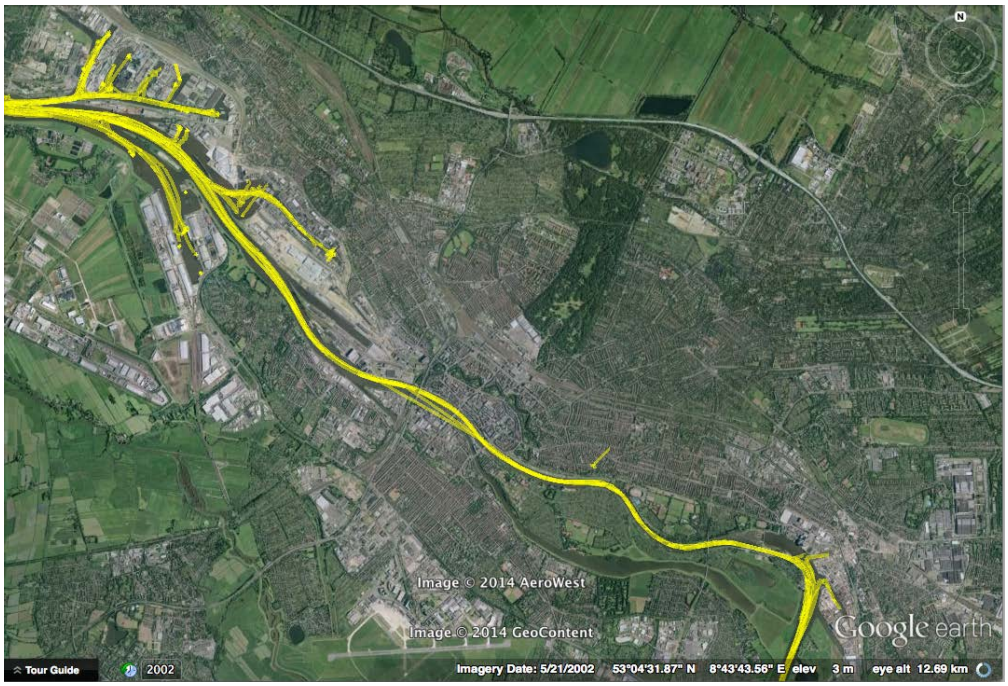


Figure B.7 – Germany, Bremen - ship traffic on the Weser.

Figure B.8 – Germany, Bremen - Harbour detail.

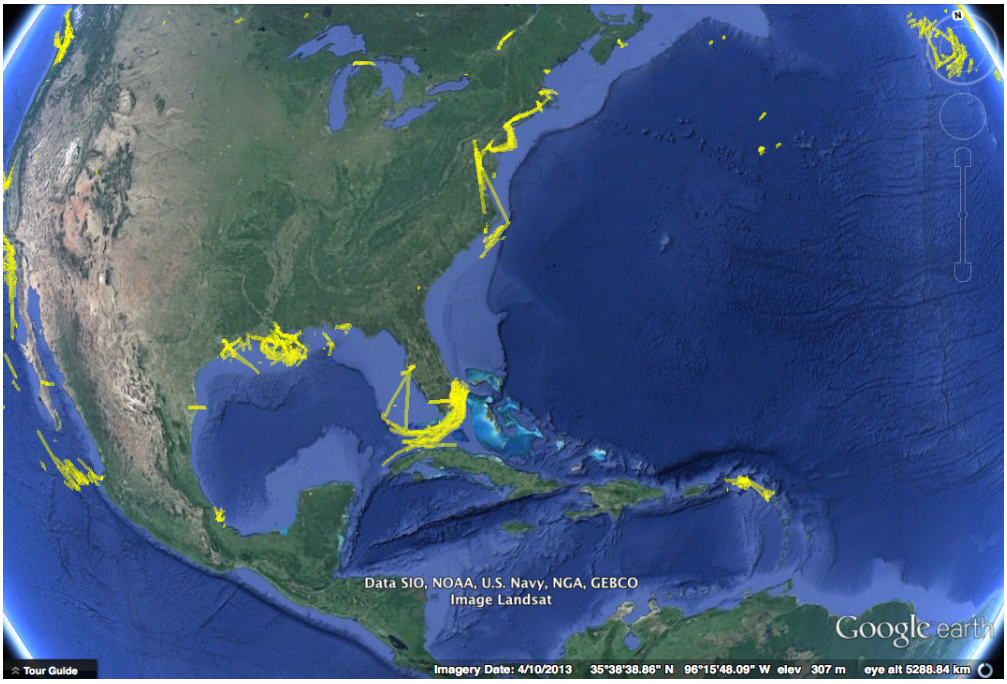


Figure B.9 – USA - coastline.

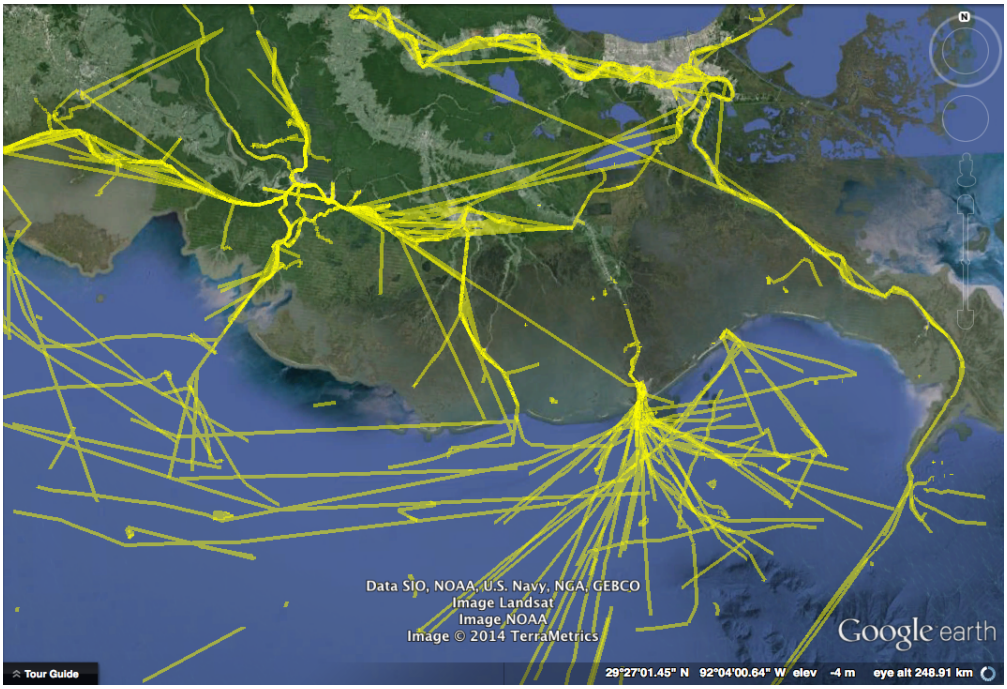


Figure B.10 – USA, New Orleans - ships following the water ways. Lines over land are interpolations between relatively large sender intervals.

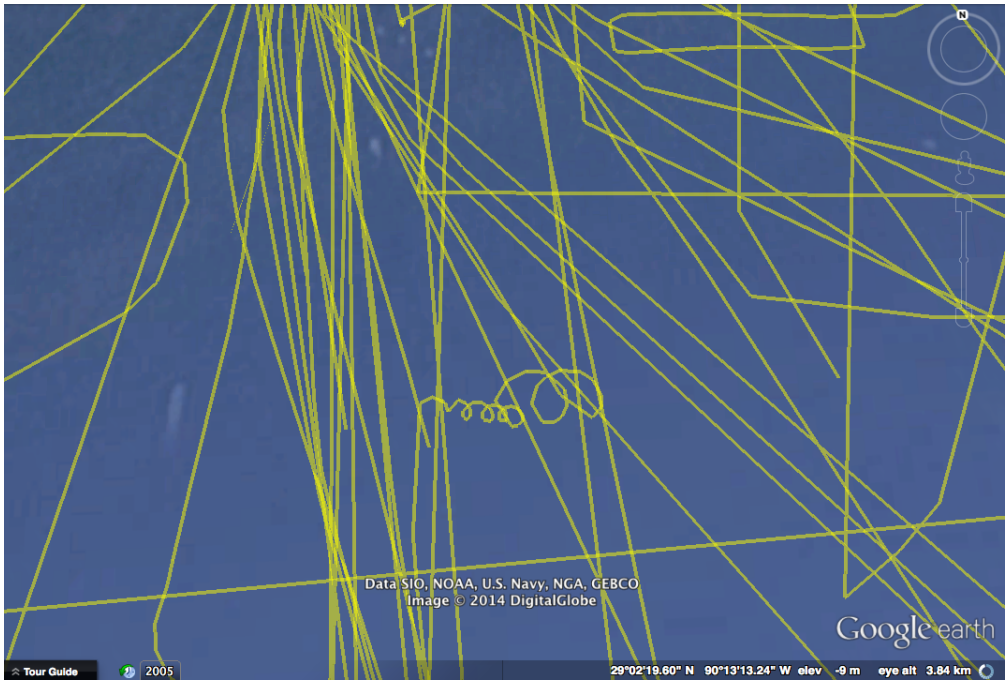


Figure B.11 – Trajectory for ship at anchor, drifting because of wind and/or current.

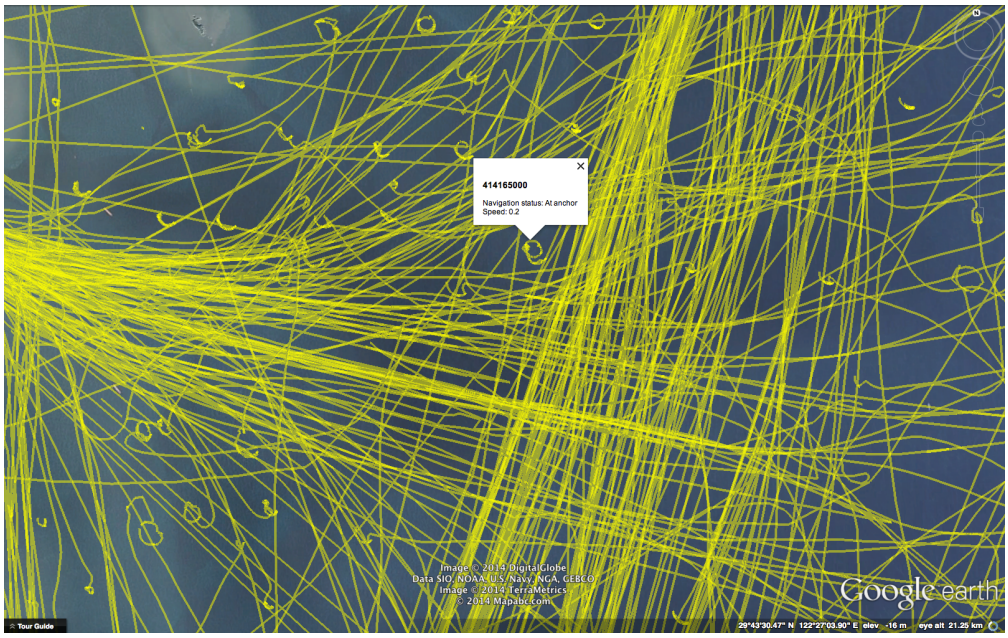


Figure B.12 – Circular lines indicate ships at anchor, drifting because of wind and/or current.