Anomaly Detection in the Surveillance Domain

Örebro Studies in Technology 50

Christoffer Brax

Anomaly Detection in the Surveillance Domain

Title: Anomaly Detection in the Surveillance Domain

Publisher: Örebro University 2011
www.publications.oru.se
trycksaker@oru.se

Printer: Intellecta Infolog, Kållered 08/2011

ISSN 1650-8580
ISBN 978-91-7668-810-6

Abstract

In the post September 11 era, the demand for security has increased in virtually all parts of the society. The need for increased security originates from the emergence of new threats which differ from the traditional ones in such a way that they cannot be easily defined and are sometimes unknown or hidden in the "noise" of daily life.

When the threats are known and definable, methods based on situation recognition can be used find them. However, when the threats are hard or impossible to define, other approaches must be used. One such approach is *data-driven anomaly detection*, where a model of normalcy is built and used to find *anomalies*, that is, things that do not fit the normal model. Anomaly detection has been identified as one of many enabling technologies for increasing security in the society.

In this thesis, the problem of how to detect anomalies in the surveillance domain is studied. This is done by a characterisation of the surveillance domain and a literature review that identifies a number of weaknesses in previous anomaly detection methods used in the surveillance domain. Examples of identified weaknesses include: the handling of contextual information, the inclusion of expert knowledge and the handling of joint attributes. Based on the findings from this study, a new anomaly detection method is proposed. The proposed method is evaluated with respect to detection performance and computational cost on a number datasets, recorded from real-world sensors, in different application areas of the surveillance domain. Additionally, the method is also compared to two other commonly used anomaly detection methods. Finally, the method is evaluated on a dataset with anomalies developed together with maritime subject matter experts. The conclusion of the thesis is that the proposed method has a number of strengths compared to previous methods and is suitable for use in operative maritime command and control systems.

Keywords: Anomaly Detection, Information Fusion, Visual Surveillance, Maritime Domain Awareness

Acknowledgements

First of all, I would like to begin by thanking my main supervisor Lars Niklasson. All your support, insightful comments, guidance and encouragement over the years have helped make this thesis into what it is. Although I had some concerns at the beginning of the thesis work, you proved that they were completely unfounded and you turned out to be a great supervisor. I would also like to thank my co-supervisor, Göran Falkman, for all his support and comments on my work.

I also want to express my gratitude to my employer, Saab AB, for supporting my Ph.D. studies, and to my colleagues Per Gustafsson, Håkan Warston, Martin Smedberg, Thomas Kronhamn, Thomas Pettersson and Tomas Planstedt for all their support, feedback and practical advice given during the thesis work. Hopefully we can continue the work of developing the field of Information Fusion at Saab.

Another group of people whose friendship I am grateful for is my fellow Ph.D. students, Fredrik Johansson, Anders Dahlbom, Maria Riveiro, Maria Nilsson, Tove Helldin and Tina Elandsson. I will miss all of you; the work will not be the same without our entertaining discussions.

I am also indebted to the Information Fusion Research Program, the University of Skövde and the University of Örebro for making this research possible.

Special thanks go to Alexander Karlsson and Rikard Laxhammar for all the interesting and fruitful discussions and for being great co-authors.

My gratitude to the Swedish Maritime Administration (Sjöfartsverket) in Gothenburg and the LFV Group (Luftfartsverket) at Landvetter Airport for their support and for providing access to subject matter experts.

Lastly, I would like to express my heartfelt thanks to my beloved family for all their encouragement and support, and for coping with a distracted husband and father over the last few years. Without your support, this thesis would never have been written.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1
Introduction

In the post September 11 era, the demand for security has increased in virtually all parts of the society. Military agencies are adapting their systems and doctrines to cope with both the traditional threats as well as new ones such as terrorism, rouge states, organized riots and so on. Civilian law enforcement agencies have received more resources and the number of privately owned security companies has increased. The European Security Advisory Board has identified a number of research topics that are considered important for future security research within the European Union [56]. In its report, four mission areas were identified: border security, protection against terrorism and organised crime, critical infrastructure protection and restoring security in case of crisis. All four missions involve extending the current capabilities of civilian and military agencies, by providing integration of existing technology as well as developing new technology. The U.S. Department of Homeland Security has also identified a number of high-priority technology needs related to increased societal security [38]. These technological needs relate to areas such as border and maritime security, cyber and information security, infrastructure protection and incident management.

The demand for increased security can be met by increasing the surveillance capability and by working proactively to minimize the possible impact from threats. The focus of this work is on increasing the surveillance capability.

The surveillance capability, that is, the capability of monitoring the behaviour of objects in an area of interest, using various types of sensors, is determined by a number of factors: (1) the size of the area, (2) the technological monitoring systems such as sensors and networks, (3) the operators and (4) the integrations between (2) and (3). Increasing one of the factors might lead to increased capabilities, but can also lead to new problems, for example, by adding more sensors, the operators working with analysing the sensor information could be overwhelmed and more operators would be needed to maintain the total capability. The reason for adding more sensors is often to increase the surveyed area. Adding sensors with overlapping coverage areas can be used

to increase the accuracy of sensor readings but puts more stress on the system responsible for correlating sensor data. If the level of automation in the system can be increased, the operators can be relieved of some of the routine tasks and focus their effort on increasing the total surveillance capability. This way, it is possible to increase the technological systems without having to increase the number of operators or to increase both the technological systems and the number of operators and obtain more capability. The focus in this work is to improve the technological systems.

The technological systems of today consist of networks of interconnected sensors which produce both soft (human intelligence) and hard (signal and communication intelligence) data [70] that needs to be communicated, stored, analysed and presented to operators. The systems produce multi-dimensional time-series information, that is, the attributes of each surveyed object are updated repeatedly with some time interval (the length of the interval depends on the type of sensor). Based on current research, it is now possible to communicate large amounts of data in real-time over long distances as well as to store the data in data management systems [33]. However, methods for automatic information analysis with the aim of producing information that can support the operators in their decision-making need to be improved [3].

The need for increased security originates from the emergence of new threats. These new, asymmetric threats differ from the traditional ones in such a way that they cannot be easily defined. Traditional threats, like the nuclear annihilation threat during the Cold War, were easy to define and followed well-known military doctrines. The new threats are much harder to define and are sometimes unknown or hidden in the "noise" of daily life [87]. Even if the properties of the threats are known, it can be hard to define all variations, for example, the smuggling of weapons is a known threat, but it is hard to define all the possible smuggling scenarios.

However, when the threats are known and definable, methods based on situation recognition [51, 35] can be used find them. On the other hand, when the threats are hard or impossible to define, other approaches must be used [121]. One such approach is to consider threats as something that is uncommon or that deviate from what can be expected. If the distribution of normal data is known, the threats could be found by using outlier detection methods. The threats will then end up outside areas of normal data. For example, if the normal speed of vehicles on Swedish roads, at any given time, is in the range of 30–110 km/h and we observe a vehicle with a velocity of 200 km/h, it can be regarded as an outlier. This might be fine for some applications, but the model is a very rough approximation of the distribution. In most real-world applications, the distribution is unknown or hard to estimate and depends on a number of joint attributes. For example, the distribution of vehicle velocities could depend on the type of road, the geographical position and the time of day. This means that it is not always possible to detect interesting behavioural anomalies just by using attributes related to the object itself, for example, kine-

matic attributes, and additinal ones describing the context in which the object appears, such as geographical or weather related attributes, might also be required [110]. These contextual attributes are used to represent what Steinberg [125] calls contextual information, and they can *constrain processing*.

To cope with an unknown distribution, *data-driven anomaly detection* approaches [85, 103, 29, 22, 87] can be used. Instead of assuming a distribution, they learn the distribution from a training dataset containing data assumed to be normal.

In the surveillance domain, operators track and analyse physical objects surveyed by sensors. If the objects behave anomalously with respect to what is normal in the domain, they could be a potential threat. However, not all detected anomalies can be considered threats. They could also arise from an incorrect model of normalcy or just be a result of non-threatening anomalous behaviour. If anomalous objects could be automatically detected and reported to the operator, the surveillance capability could be increased by allowing the operators to focus on the subset of objects that have the highest possibility of being threatening.

According to the Department of Homeland Security [3], *anomaly detection* is one of many enabling technologies for Maritime Domain Awareness (MDA). MDA is important for all maritime authorities and involve comprehending all maritime activities and their possible impact on security, safety, environment and the economy. The goal of anomaly detection is to find "objects that are different from most other objects" [132]. The anomaly detection method shall "discover the real anomalies and avoid falsely labelling normal objects as anomalous" [132]. In essence, a well performing anomaly detection method should have a high detection rate without too many false alarms. Portnoy et al. [105] made the following definition in the domain of network intrusion detection:

> "Anomaly detection approaches build models of normal data and then attempts to detect deviations from the normal model in observed data." (p.2)

Based on this definition, one way of detecting anomalies is to use normal data to build a model that describes what is considered to be normal. This model can then be used to classify new data as normal or anomalous.

1.1 Aims and Objectives

Previous research on anomaly detection in the surveillance domain [85, 22, 87, 48, 108, 77, 68, 121] does not put much emphasis on the inclusion of contextual information and expert knowledge about the domain, when building the models of normalcy. The knowledge of experts is important for two reasons; it could increase the performance of the detection and it could also increase the operators trust in the system, when the detection is based on information and parameters known by the operator. The context plays an important role when

trying to define what is normal. For example, the traffic situation in a city is related to both the time of day and the day of the week. If we do not include this information in the models, the system will not be able to tell the difference between a traffic jam caused by morning traffic and a traffic jam caused by an accident during the weekend. A traffic accident in the morning traffic will, however, be very hard to find, because it is difficult to qualitatively distinguish between traffic jams caused by accidents and those caused by many vehicles on the road.

This leads to the aim of the research presented in this thesis. The aim is to:

> Develop an accurate, data-driven anomaly detection method that is transparent, computationally efficient, can incorporate contextual information and expert knowledge, can handle joint attributes and use time-series information to detect anomalies in the surveillance domain.

The aim entails questions such as: Can the accuracy of an anomaly detection method used in the surveillance domain be improved by including contextual information in the representation and the normal model? How can joint attributes with unknown distributions be modelled efficiently? Can the accuracy of an anomaly detection method be improved by combining anomaly classifications over time? How can expert knowledge be included in an anomaly detection method?

In order to fulfill the aim, the following objectives have been identified:

O1. Characterise the properties of the surveillance domain that are important to anomaly detection.

Before a suitable method for anomaly detection can be found, the properties of the surveillance domain must be identified. These properties affects the requirements for the anomaly detection method.

O2. Review and analyse existing methods for anomaly detection in the surveillance domain.

There is a number of anomaly detection methods commonly used in the surveillance domain. The objective includes a literature review and the identification of shortcomings in previous approaches based on domain requirements and domain properties. Another important part of this objective is to identify important properties for quantitatively and qualitatively evaluating anomaly detection methods in the surveillance domain.

O3. Propose and implement an anomaly detection method based on the result of the literature review.

Based on the result of the first two objectives, a new anomaly detection method will be proposed. The new method will extend previous methods to address some of the identified shortcomings from objective 2.

O4. Evaluate the proposed method on datasets from different application areas of the surveillance domain.

In order to be able to assess the applicability of the proposed method in the surveillance domain, it will be evaluated on a number of datasets from different application areas of the surveillance domain, such as, indoor and outdoor video surveillance, maritime surveillance and land transportation.

O5. Compare the proposed method to other methods previously used in the surveillance domain.

To be able to assess whether the proposed method has any merits compared to previous methods, it must be evaluated and compared to previously used methods in the surveillance domain. This comparison will be done both by quantitatively evaluating properties of the methods as well as qualitatively evaluate the performance of each method on the same datasets.

1.2 Research Methodology

A number of different research methodologies are used in this thesis. The first two objectives are addressed by conducting two *literature surveys* [91]. Papers published in conference proceedings and journals on Information Fusion, Sensor and Signal processing, and Video Image Analysis are used as input to the surveys. The thired objective is addressed by using *implementation* [19]. The result of the implementation is an experimental platform that is incrementally extended to cope with different anomaly detection problems. The experimental platform is used to conduct *empirical experiments* [41] in order to address objectives four and five. Objective four is addressed by both *interviewing* subject matter experts and by empirical experiments based on the result of the interviews.

It can sometimes be hard to make a quantitative comparison between anomaly detection methods. This is a result of two problems in the Information Fusion (IF) community and other related communities. The first problem is the lack of standard benchmarking datasets. In the data-mining domain, the famous UCI datasets [5] help researchers to quantitatively compare algorithms. In the IF community, there are no such datasets. Instead, each researcher uses their own dataset, and it is not uncommon for the datasets to be proprietary and owned by a company or agency. This often means that other researchers cannot make a direct comparison of performance between algorithms without implementing the algorithms and running experiments on their own dataset. The second problem is that many publications in the area lack the degree of detail that is needed to implement and evaluate the proposed algorithms.

In this work, a number of anomaly detection methods are qualitatively compared with respect to important properties for anomaly detection in the sur-

veillance domain. Our proposed method is quantitatively evaluated on a number of real-world datasets with added anomalies and compared to two methods based on Gaussian Mixture Models and Kernel Density Estimators.

1.3 Delimitations

The focus of this work is the development and evaluation of anomaly detection methods for finding anomalies in the surveillance domain. In a complete system, the anomaly detection system should be used together with a number of other systems (e.g. sensors, communication, trackers and Command & Control) and users. The users have an important role in the complete system and the interaction with between the system and the users is of great significance. This is, however not in the scope of the thesis and the problems related to this topic are not addressed in any depth. The life cycle aspect of anomaly detection systems is another area that is outside the scope of this work.

1.4 Scientific Contributions

The main contribution in this work is a novel method for unsupervised anomaly detection with the ability to incorporate kinematic data, contextual information and domain expert knowledge into the same model. The method includes a representation scheme, normalcy modelling and algorithms for the detection of anomalies.

The method has very low computational requirements both when building the normal model and when detecting anomalies. There is also the possibility to update pars of the normal model without rebuilding it from scratch.

The main contributions of this thesis are:

- A new method for anomaly detection called *The State-Based Anomaly Detection* (SBAD) method.
- An evaluation of the method on a number of surveillance datasets.
- A performance comparison between the SBAD method and two methods based on Gaussian mixture models and Kernel Density Estimators.
- An extension to the State-Based Anomaly Detection (SBAD) method for detecting anomalous state transitions and anomalous stops.
- A study of real-world anomalies for the evaluation of anomaly detection methods and a feasibility assessment of the use of the SBAD method in the maritime domain.

1.5 Publications

This section lists relevant publications and their contribution to the thesis.

1. Niklasson, L., Riveiro, M., Johansson, F., Dahlbom, A., Falkman, G., Ziemke, T., Brax, C., Kronhamn, T., Smedberg, M., Warston, H. and Gustavsson P.M., (2007) A Unified Situation Analysis Model for Human and Machine Situation Awareness, Lecture Notes in Informatics, pp 105–110, Köllen Druck+Verlag GmbH, Bonn. ISBN 978-3-88579-206-1.
   This paper jointly written by some of the members of our research group aims to reach a common view of how different research projects are related to each other and to concepts used in the Information Fusion community. The paper introduces a unified model that is used to describe how the work in this thesis relates to other work in the field of Information Fusion.

2. Brax, C., Niklasson, L., and Smedberg, M. (2008) Finding behavioural anomalies in public areas using video surveillance data. Proceedings of the 11th International Conference on Information Fusion, Cologne, Germany, June 30-July 3, pp. 1655–1662. ISIF - IEEE. ISBN: 978-3-00-024883-2.
   This paper introduces the State-Based Anomaly Detection method, including a representation scheme, normalcy modelling and detection of anomalies. The method is evaluated on a real-world video surveillance dataset. The paper contributes to objectives 1, 2, 3 and 4 of the thesis.

3. Brax, C., Laxhammar, R., and Niklasson, L., (2008), Approaches for automatically detecting behavioural anomalies in public areas using video surveillance data, In Proceedings of SPIE Europe, Vol. 7113, pp. 711318-1–711318-12, 15–18 September 2008, Cardiff, Wales. ISBN: 978-0-8194-7352-3, DOI: 10.1117/12.800095.
   The material in this paper extends the work in (2) by comparing the proposed method with another commonly used method. Both methods are evaluated on the same dataset. The paper contributes to objectives 3, 4 and 5 of the thesis. Co-author Rikard Laxhammar contributed with the design and implementation of the anomaly detection approach based on Gaussian Mixture Models (GMM).

4. Niklasson, L., Riveiro, M., Johansson, F., Dahlbom, A., Falkman, G., Ziemke, T., Brax, C., Kronhamn, T., Smedberg, M., Warston, H., and Gustavsson., P. M. (2008) Extending The Scope of Situation Analysis, Proceedings of the 11th International Conference on Information Fusion, Cologne, Germany, June 30-July 3, pp. 454–461. ISIF - IEEE. ISBN: 978-3-00-024883-2.
   This paper extends the work in (1) by introducing a collaborative dimension in the model. The paper contributes to the background of the thesis.

5. Brax, C. and L. Niklasson. (2009), Enhanced Situational Awareness in the Maritime Domain: An Agent-based Approach for Situation Management. In Proceedings of SPIE, Vol. 7352. pp. 735203-1–735203-10. Orlando, Florida, USA, 13–17 April 2009. ISSN: 0277-786X (print), ISBN: 9780819476180, DOI: 10.1117/12.81847.
   The method in (2) is evaluated on maritime vessel data from an airborne early-warning radar system. The paper contributes to objectives 3 and 4 of the thesis.

6. Brax, C., L. Niklasson, and Laxhammar, R., (2009), An ensemble approach for increased anomaly detection performance in video surveillance data. Proceedings of the 12th International Conference on Information Fusion, pp. 694–701, Seattle, USA. ISIF. ISBN: 978-0-9824438-0-4.
   In this paper the method proposed in (2) is extended. The analysis now includes three aspects of object behaviour. The method is evaluated on a large and complex real-world dataset collected by a video-surveillance system at an airport. The method is evaluated against another method and the results are compared to the results of fusing the output of both methods together. The paper contributes to objectives 1, 2, 3, 4 and 5 of the thesis. Co-author Rikard Laxhammar contributed with the design and implementation of the anomaly detection approach based on Gaussian Mixture Models (GMM).

7. Fooladvandi, F., Brax, C., Gustavsson, P., and Fredin, M. Signature-based activity detection based on Bayesian networks acquired from expert knowledge. Proceedings of the 12th International Conference on Information Fusion, pp. 436–443, Seattle, USA. ISIF. ISBN: 978-0-9824438-0-4.
   This paper presents a signature-based approach for finding interesting situations, which is a complementary approach to unsupervised anomaly detection. The paper contributes to the background of the thesis.

8. Brax, C., Fredin, M., (2009), Increased transportation security by using automatic detection of anomalous truck behaviour. Proceedings of the 16th World Congress and Exhibition on Intelligent Transport Systems and Services, Stockholm, Sweden.
   The paper reports on initial experiments using unsupervised anomaly detection for finding anomalous truck behaviour. The work is part of a larger project that deals with increased security in intermodal transportation chains. The paper contributes to objectives 3 and 4 of the thesis.

9. Brax, C., Niklasson, L., (2009), An approach for increased supply chain security by using automatic detection of anomalous vehicle behaviour, Proceedings of the 6th International Conference on Modeling Decisions for Artificial Intelligence, pp. 165-176, Awaji Island, Japan. ISBN: 978-84-00-08851-4. [CD-ROM]

This paper extends the experiments proposed in (8) and contributes to objectives 1, 2, 3 and 4 of the thesis.

10. Brax, C., Karlsson, A., Andler, S. F., Johansson, R., and Niklasson, L. (2010), Evaluating Precise and Imprecise State-Based Anomaly Detectors for Maritime Surveillance, Proceedings of the 13th International Conference on Information Fusion, IEEE. ISBN: 978-1-9824438-1-1.
This paper compares a three approaches for fusing anomaly classifications over time. It is based on the approach introduced in (6) and adds two new approaches based on bayesian evidence theories for uncertainty management. The proposed method is evaluated and compared to two other anomaly detection methods. The paper contributes to objectives 3, 4, 5 and 6 of the thesis. The author and Alexander Karlsson have made equal contributions to this publication. The author contributed with the anomaly detection method that was extended in the paper, as well as the overall design of the experiments and datasets.

11. Brax, C., Karlsson, A., Niklasson, L., (Submitted), An Empirical Study of Anomaly Detection Methods for Increased Situation Awareness in the Maritime Domain, Submitted to the Journal of Advances in Information Fusion.
This paper includes a detailed analysis of the proposed method on a real-world maritime dataset developed together with domain matter experts. The paper also proposes a simulation-based method for setting appropriate thresholds for the anomaly detection method. The paper contributes to objectives 3, 4, 5 of the thesis.

1.6 Thesis Outline

The thesis is organized accordingly: following the introduction, Chapter 2 presents the background to the thesis. The background includes information regarding Information Fusion, Anomaly Detection and other related areas of research. In Chapter 3, the surveillance domain is analysed and a number of properties are identified. This chapter also includes an analysis of previous anomaly detection methods used in the surveillance domain. The State-Based Anomaly Detection (SBAD) method is introduced in Chapter 4, together with an evaluation on datasets from an outdoor video surveillance scenario and an airborne radar scenario. Chapter 5 presents an extension of the SBAD method which is evaluated on data from land transportation as well as on data from an indoor video surveillance scenario. In Chapter 6, the temporal aspects of the SBAD method are more thoroughly evaluated and analysed on a dataset developed together with maritime subject matter experts. The thesis concludes in Chapter 7, with a summary, conclusions and a discussion about a number of areas for future work.

Chapter 2
Background

This chapter presents the background to the work in the thesis and introduces some of the concepts that are used throughout the thesis. In Section 2.1, the domain of Information Fusion is described. This section is used to put anomaly detection into the context of technical support systems used for aiding human decision makers. Section 2.2 introduces the reader to the general anomaly detection problem and describes a number of classes of methods previously used for anomaly detection. In Section 2.3, a brief introduction to uncertainty management is presented. This introduction can be used to better understand the work presented in Chapter 6.

2.1 Information Fusion

Information fusion is a multi-disciplinary research field in which researchers develop methods and algorithms for combining data from different sources to perform inferences that would be hard to do with information from a single source [69]. Dasarathy [45] defines information fusion as:

> "Information fusion encompasses the theory, techniques, and tools conceived and employed for exploiting the synergy in the information acquired from multiple sources (sensor, databases, information gathered by human etc.) such that the resulting decision or action is in some sense better (qualitatively and quantitatively, in terms of accuracy, robustness etc.) than would be possible, if these sources were used individually without such synergy exploitation."
> - Dasarathy [45]

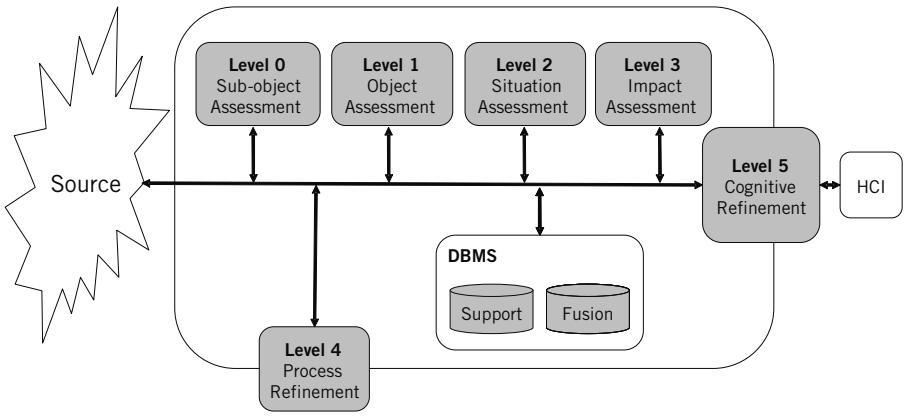


Figure 2.1: The JDL model. Adapted from Hall and McMullen [69].

Information fusion is sometimes referred to as data fusion[1]. To obtain a better understanding of what processes are involved in data and information fusion, the Joint Directors of Laboratories (JDL) data fusion sub-panel developed a model called the JDL model [69]. This model has been subjected to a number of revisions over the years [126, 21, 95]. Hall and McMullen [69] suggest the version shown in Figure 2.1.

The JDL model defines the processes needed in an information fusion system. However, it does not describe how the processes interact or how they should be implemented. The model describes processes for assessment of sub-objects (or signals), objects, situations and impacts. It also includes two refinement processes that deal with improvements to the other processes and with feedback from the user [69]. According to [131], the processes in Level 0 and Level 1 of the JDL model can be regarded as sensor fusion processes while the higher levels are considered to be information fusion processes.

Hall and McMullen [69] describe the levels in the JDL model as:

Level 0 - Sub-object Assessment Also referred to as source pre-processing or signal assessment. This level includes processes for the pre-processing of data from sensors and databases, such as bias corrections, as well as unit conversions, filtering and feature extraction.

Level 1 - Object Assessment This level includes processes for combining data from different sensors to obtain estimates of an object's location, mo-

[1]The research field has traditionally been called data fusion. In recent years the name information fusion has been increasingly used instead of data fusion. Both international conferences and journals use the name information fusion. This might be an effect of the inclusion of more high-level problems (e.g. situation and impact assessment) and "softer" aspects (e.g. human intelligence, cognitive aspects of decision-making) into the research field.

tion, attributes, identity and characteristics. Common level 1 processes are target tracking (e.g. Kalman filtering and multi-hypothesis tracking) and pattern recognition for identity determination.

Level 2 - Situation Assessment Level 2 processes include assessment of relations between objects and the environment to obtain a better understanding of the current situation. Common tasks are object aggregation, event detection and multi perspective reasoning. It is common to use methods based on artificial intelligence and automated reasoning at this level.

Level 3 - Impact Assessment Level 3 processes deal with projections about possible future situations and hypotheses about the current situation to determine potential impacts from an evolving situation. This level includes threat evaluation, risk assessment, probable courses of actions and opportunities. Methods from artificial intelligence, automated reasoning, statistical estimation and predictive modelling are often used at this level.

Level 4 - Process Refinement The processes at level 4 monitor the on-going information fusion processes and try to optimize the algorithm performance and the utilization of sensors. The processes at level 4 feed information back to all levels from 0 to 3.

Level 5 - Cognitive Refinement Level 5 processes monitor the interaction between the information fusion system and the human decision-maker and were introduced by Hall et al. [71]. This enables human-in-the-loop information fusion.

Anomaly detection can be regarded as a level 2 or level 3 process depending on the time frame. If level 1 information from the current time frame is used, the anomaly detection is a level 2 process and the output can be presented to a human decision maker or used as input to level 3 processes. If the time frame is in the future the anomaly detection can be regarded as a level 3 process, c.f. prediction of anomalies in Section 7.2.7.

2.1.1 The OODA Loop

The OODA (observe, orient, decide and act) loop, depicted in Figure 2.2, has been used to describe the iterative and cyclic concepts of tactical command and decision-making [69]. The OODA loop was originally developed by an American aviator named John Boyd [24], during the Korean War, and was used to describe why the American fighters were so successful against the Korean Air-Force. According to Azuma et al. [16], the general strategy for defeating an enemy is “getting inside his OODA loop”. This is usually done by iterating your own loop faster than the enemy can iterate through his loop. If this can be accomplished, the enemy’s awareness of the situation is not up to date and

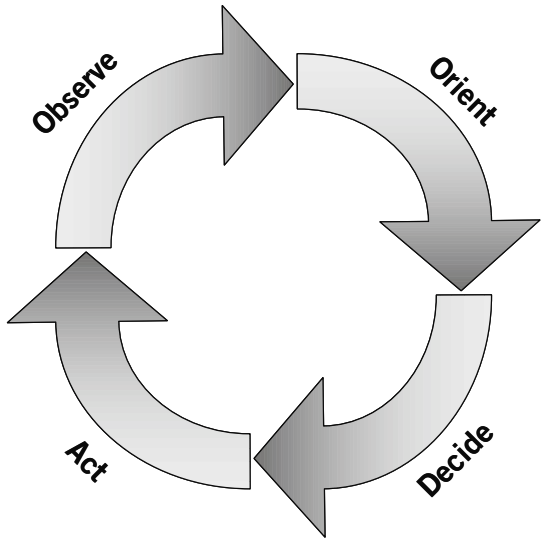


Figure 2.2: The OODA Loop (Adapted from Boyd [24]).

the situation assessments that are the basis of the enemy's decisions will be old and inaccurate.

The model used an abstract perspective with only four steps (Hall and McMullen [69]):

Observe: Collect data from humans and sensors to find information about a situation.

Orient: Relate the information to current knowledge to assess the situation. How did past decisions affect the situation?

Decide: Based on the assessment in the orient step, decide on a suitable action while considering the likelihood of possible hypotheses and the consequences for each possible hypothesis.

Act: Carry out the decision by performing necessary actions. These actions might be collect additional data, order a military offensive, tune sensor parameters, request additional modelling of the world, or other activities.

Over the years, there has been some criticism of the OODA loop, for example, Bryant [34] who argues that the OODA loop does not describe proactive decision-making at all, but instead only describe reactive decision-making. This means that the decision maker only bases decisions on the result of the orientation step and not on long-term plans. Bryant [34] also argues that the OODA loop is based on 50-year old cognitive theories and should therefore be revised.

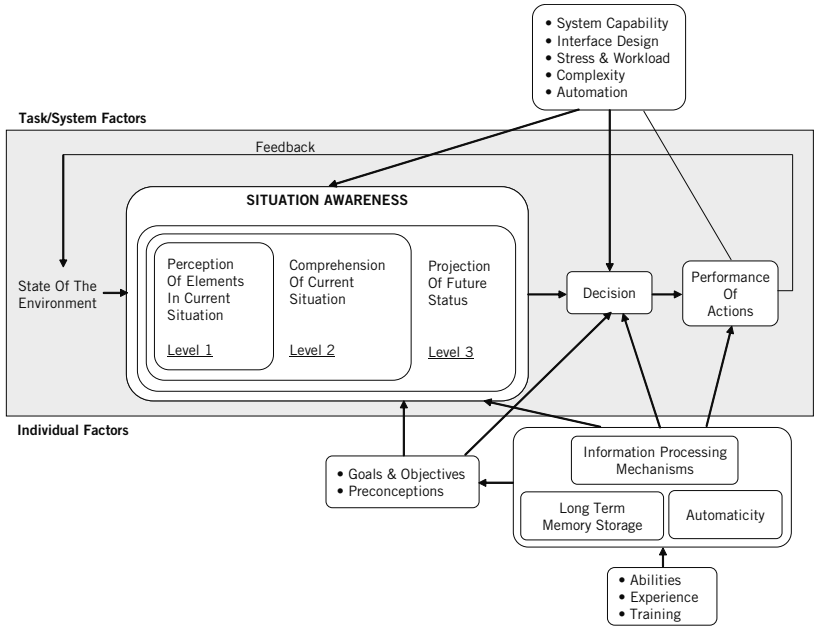


Figure 2.3: Endsley's Situation Awareness model (adapted from Salerno et al. [119]).

2.1.2 Situation Awareness

The concept of Situation Awareness (SAW) is important in dynamic decision-making. Endsley [53] proposes a general definition of SAW:

> "Situation awareness is the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future."
> - Endsley [53]

Endsley's view on SAW concerns how to combine, interpret, store and retain information [53]. In Endsley's model (see Figure 2.3), SAW is divided into three levels: perception, comprehension and projection. At the perception level, attributes and dynamics of the elements in the environment are perceived. At the comprehension level, multiple pieces of information are integrated and their relevance to the decision maker's goals is determined. At the projection level, future events are predicted. This ability allows the decision maker to take timely decisions [114].

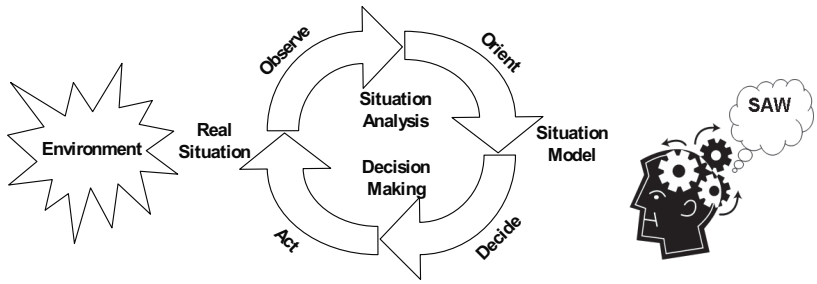


Figure 2.4: The relation between situation analysis, decisoin making and situation awareness (adapted from [114]).

2.1.3 Situation Analysis

According to Endsley [53], the state of SAW is separated from decision-making. SAW can be described as "the decision maker's internal model of the state of the environment" - Roy [114]. The decision maker uses the model as a basis for decisions about the situation. As illustrated in Figure 2.4, SAW is a prerequisite for decision-making.

In the perspective of the OODA loop, Roy defines situation analysis (SA) as: "a process, the examination of a situation, its elements, and their relations, to provide and maintain a product, i.e., a state of SAW, for the decision maker." - Roy [114]. As seen in Figure 2.4, the SA process involves the understanding of the world part in the OODA loop. According to Roy [114], the SA process takes the real situation in the environment and uses it to set up a mental representation of the real situation in the head of the decision maker. The mental representation is also referred to as the situation model.

2.1.4 The $(SAM)^2$ model

There are two aspects of decision support systems, the technological and the human [99]. These two aspects are merged together into a single model called the unified *situation analysis model for semi-automatic, automatic and manual decision support* $(SAM)^2$. The idea behind the model is that it should be possible to apply to decision support systems with an arbitrary degree of automation. The model is depicted in Figure 2.5. While the left side reflects the technological aspect and describes the different levels of the JDL model, the right side reflects the human aspects and is related to Endsleys situation awareness model. The two sides are connected to a human-computer interaction interface. In an automatic situation analysis system, only the left side is active and, in a corresponding manual system, only the right side is active. In semi-automated systems (which includes most of today's systems), the processes in the two sides

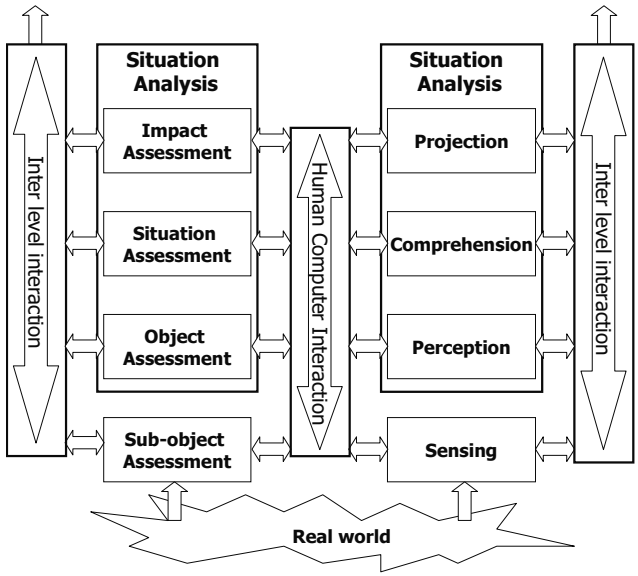


Figure 2.5: The unified model for Situation Analysis $(SAM)^2$ [99].

interact at various levels. The model can be used to relate different information fusion problems to each other.

An anomaly detection system used in the thesis is an example of a semi-automatic decision support system, which is aimed at helping the operator to focus on the right information at the right time.

2.1.5 Situation Management

Jakobson et al. [76] state that Situation Management (SM) is a research discipline that deals with: (1) aspects related to the meaning of situations, (2) methods for reasoning about situations and (3) action planning. Jakobson et al. list a number of other disciplines that are related to situation management, such as Artificial Intelligence (AI), Semantic Web, Sensor Networks, Multi-Agent Systems (MAS), Information Fusion, Self-Organizing System and Human Factors. Situation Management can be defined as:

> "... a synergistic goal-directed process of (a) sensing and information collection, (b) perceiving and recognizing situations, (c) analyzing past situations and predicting future situations, and (d) reasoning planning and implementing actions so that desired goal situation is reached within some pre-defined constraints" – Jakobson et al. [76]

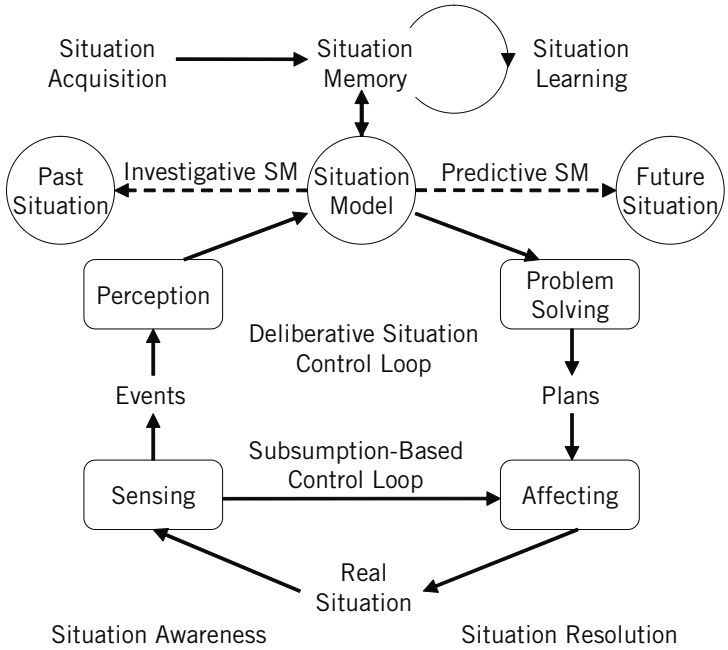


Figure 2.6: The general process loop of Situation Management (Adapted from [76]).

Based on the definition, the SM process starts with a goal. Depending on the goal, one or more of the three aspects investigative, control and predictive can be applied of the process. Jakobson et al. describe the aspects as follows:

- Investigative SM: A retrospective analysis of a situation to determine why a certain situation occurred.
- Control SM: Keeps track of the current situation.
- Predictive SM: Predicts possible future situations.

The three aspects of SM are depicted in Figure 2.6. The control loop is built on the four processes: sensing, perception, problem solving and affecting. The subsumption-based control is based on direct reaction to changes in the sensory information while the deliberative situation control involves much more analysis and reasoning. Jakobson et al. [76] argue that deliberative situation control is often vital in handling complex dynamic systems.

Besides the general process loop, Jakobson et al. also define some constituents of a general framework for SM. These constituents include a number of structural objects, such as entities, attributes, classes of entities and relations

between entities as well as dynamic components such as situations and events. The rationale behind the framework is to be able to model the processes in SM; which is referred to as Situation Modelling.

An entity is something real or abstract which has significance in the modelled domain. An entity has a number of attributes and, if a set of entities has the same attributes, it can be defined as an abstract entity class. The attributes which define an entity are represented by a number of properties, such as attribute name, attribute value, default value and so on. An attribute value is a triplet of the actual value, uncertainty information and a time-stamp. Jakobson et al. define a relation as "...a mental abstraction of linking a certain number, very often two, entities together". Relations can also have a number of attributes and are in this sense very similar to entities. The two last components in Situation Modelling are the dynamic components called situations and events. A situation has a duration, i.e., a start and end time. Jakobson et al. identify three different classes of situations: entity-based relations, relational entity-based situations and relational situations. An entity-based situation is a collection of entity states with constant values for the duration of the situation. A relational entity-based situation is similar to the entity-based situation with the difference that it uses relational states instead of entity states. Relational situations are defined as the state of the relation between two entities and do not relate to the attribute values, as for the two preceding relations. The last term defined by Jakobson et al. is the event. An event is a transition from one state to another or from one situation to another. Events can be considered as an instant at a specific point in time or as a period of duration.

2.2 Anomaly Detection

The goal of anomaly detection is to find "objects that are different from most other objects" [132]. Anomalous objects are also referred to as outliers, novelties, noise, deviations or exceptions [135]. In this work, we use the terms "anomaly" and "anomaly detection". The concept of anomaly detection is very vague; it does not define the detection approach or what to detect [52]. Anomaly detection techniques have been tailored to solve problems in many different domains, such as intrusion detection [103, 79, 89, 105, 128], fraud detection [43, 58], database optimization [92], climate studies [60], software testing [129, 63], visual surveillance [29, 28, 30, 48] and maritime surveillance [85, 22, 87, 26, 112, 108]. Chandola et al. [40] argue that anomaly detection is important due to the fact that the detected anomalies often correspond to actionable information within the application domain. For example, an anomalous communication pattern might be the result of an intrusion, anomalous program flows could be the result of a software bug, while the anomalous behaviour of vessels could indicate illegal activities or an accident.

Although anomaly detection has been studied for a long time, many methods have been developed for specific problems or domains [40]. Anomaly

detection algorithms must often be tailored for the specific properties of the data and the domain in which they are to be used. This is very similar to data mining problems where a large number of methods are used to solve a wide range of problems. In fact, anomaly detection can be seen as an approach for solving descriptive data mining tasks [132].

2.2.1 The General Anomaly Detection Problem

There are many definitions of what constitutes an anomaly and the task of anomaly detection. Tan et al. [132] use the following general definition of anomaly detection and anomalies:

> "Anomaly detection is the task of identifying observations whose characteristics are significantly different from the rest of the data. Such observations are known as anomalies or outliers." – Tan et al. [132]

Furthermore, Tan et al. [132] state that the goal of an anomaly detection algorithm is "to discover the real anomalies and avoid falsely labelling normal objects as anomalous". In essence, a well performing anomaly detection algorithm should have a high detection rate without too many false alarms. Chandola et al. talk about patterns in data instead of observations when they define what constitutes an anomaly:

> "Anomalies are patterns in data that do not conform to a well defined notion of normal behaviour". - Chandola et al. [40]

On the basis of this definition, it follows that a well-defined notion of normal behaviour is a key component in anomaly detection, together with a mechanism for assessing how well patterns conform to the notion of normal behaviour. Consider the example in Figure 2.7, most of the data is located in the regions R_1 and R_2 and can be considered to describe the normal behaviour of the data. The data points a_1, a_2 and a_3 do not conform with the normal behaviour and can be considered anomalous.

Portnoy et al. [105] include more details on the notion of normal behaviour in their definition of anomaly detection:

> "Anomaly detection approaches build models of normal data and then attempts to detect deviations from the normal model in observed data." (p.2)

This definition describes a model-based approach for detecting deviations. Data assumed to be normal is used to build a model that describes normal behaviour, which can be used to classify new data as normal or anomalous. While this definition explains one way of doing anomaly detection, it does not provide any details about how to represent the data, how to create the normal model,

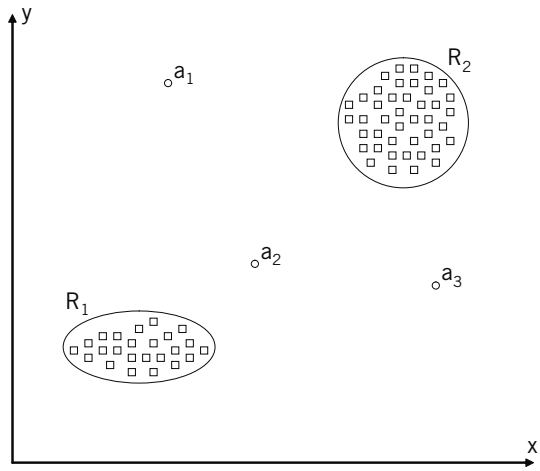


Figure 2.7: An example of anomalies in a two-dimensional dataset. R_1 and R_2 are regions with normal data points and a_1, a_2 and a_3 are data points outside the normal regions and can be considered anomalies.

or how to use the representation and the normal model to detect anomalies [28].

Chandola et al. [40] argue that the main difference between an anomaly and noise in data is that the anomalies are interesting for an analyst, while noise is unwanted and complicates the analysis. Many anomaly detection methods can also be used for noise removal.

2.2.2 Challenges in Anomaly Detection

At a high level, anomaly detection is just about defining the regions of normal data and then using these regions to classify everything outside them as anomalies. However, there are a number of factors that make this approach very hard to implement in practice. Chandola et al. [40] have identified the following challenges:

1. Defining the regions of normal data in such a way that it captures all possible kinds of normal behaviours is very hard, partly due to the fact that the boundaries of the regions are seldom precise.

2. If the anomalies are the result of malicious activity, the antagonists often change their behaviour to blend in and hide within the normal behaviour of others, which makes it harder to define the normal regions and their boundaries.

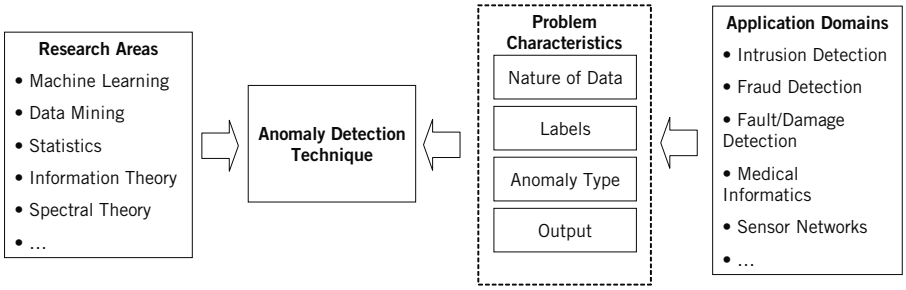


Figure 2.8: Key aspects to consider when developing an anomaly detection technique. Adapted from Chandola et al. [40].

3. In many domains the behaviour of objects evolves over time. This means that the definition of what constitutes normal behaviour must be constantly refined to reflect the normal behaviour of objects in the domain.

4. What constitutes an anomaly varies for different domains. In some domains a small deviation might be normal, but in others that small deviation indicates an important anomaly. This makes it hard to apply a domain specific method in another domain.

5. The availability of datasets with labelled data for training and evaluation is often an issue, especially when using recorded, real-world datasets.

6. Real-world datasets often contain noise that is hard or impossible to remove, mainly because the noise is very similar to the actual anomalies. The presence of noise could have a negative impact on the performance of the anomaly detection method.

According to Chandola et al. [40], the challenges above make the general anomaly detection problem very hard to solve, and most of the current methods address the problem under a number of constraints. These constraints are induced by the properties of the available datasets, the requirements of the domain, the nature of the anomalies to be detected and so on. Methods and concepts from various domains have been adapted by researches to be used for solving anomaly detection problems. See Figure 2.8 for a pictographic description of the key components in an anomaly detection technique.

Furthermore, Tan et al. [132] identify the following issues related to anomaly detection.

Number of features to use when defining an anomaly: An object can be anomalous on the basis of a single attribute or a combination of multiple ones. It might be normal to be 1 meter tall (children) or weigh 150 kg,

but the combination of being 1 meter tall and weighing 150 kg could be anomalous. If the data includes dependencies between features, this must be handled by the anomaly detection method, which can pose a problem, if the dimensionality of the data is high. In this thesis, features with dependencies are referred to as *joint features*, i.e., features that must be combined to be usefull.

Global/local perspective: An object can be anomalous with respect to all objects of a certain type, but not compared to other objects of the same sub-type. This means that the context is sometimes very important. A vehicle driving at 200 km/h might be very unusual compared to all vehicles, but not compared to Formula 1 racing cars.

Anomaly measure: When presenting anomalies to an analyst it might be suitable to use a binary classification, i.e., whether an object is anomalous or not. In reality, it is common for objects to be anomalous to different degrees. Some objects may be extremely anomalous while others are almost normal. Hence, the ability to set an anomaly score or degree of anomaly is a desirable property of an anomaly detection method. This score can be used together with a threshold to obtain a binary classification if that is required.

Evaluation: The evaluation of anomaly detection methods is vital. Without a proper evaluation, it is impossible to assess wheather the method really aids the analyst. If the objects in the dataset have a label indicating whether they belong to the class of normal or anomalous objects, common data mining measures can be used to measure the performance of a method. Since most real-world datasets have an imbalance between objects labelled normal and whose labelled anomalous, the use of measures like precision, recall, false positive rate is more appropriate than accuracy. When class information is missing, the evaluation becomes much harder.

Efficiency: The computational cost of doing anomaly detection can vary greatly depending on which method is used. The efficiency can be measured in two different stages, when creating the model and when applying it to detect anomalies. Domain requirements might constrain the amount of computation time available in each stage.

2.2.3 Different Aspects of the Anomaly Detection Problem

As can be seen in Figure 2.8, there are a number of different aspects to consider when dealing with anomaly detection problems. Chandola et al. [40] identified nature of the data, type of anomaly, class labels and output as the most important ones and they will impact the formulation of the specific anomaly detection problem.

2.2.3.1 Nature of Input Data

According to Chandola et al. [40], the input data has a direct impact on the choice of anomaly detection technique. The input generally consists of a collection of data instances which are sometimes also referred to as objects, records, points, vectors, patterns, events, samples, observations or entities (Tan et al. [132], Chapter 2). Each instance is defined by a number of attributes which can also be called variables, characteristics, features, fields or dimensions [40]. According to Chandola et al. [40], the attributes can belong to one of the following data types: binary, categorical or continuous. A data instance can have one or more attributes of the same or different data types. Chandola et al. [40] state that the characteristics of the attributes in the data have a major impact on the choice of anomaly detection technique. For example, when using a statistical technique, the underlying model is dependent on whether the attributes in the data are categorical or continuous. Likewise, the distance measure used in a nearest neighbour-based technique is dependent on the nature of the attributes in the data.

Another way of categorising input data is to regard the relationship between data instances [132]. Many existing anomaly detection techniques only deal with single point data. In general, there are a number of possible relations between data instances, such as sequential data, spatial data and graph data. In a sequential dataset, the data instances are ordered linearly. Examples of sequential datasets are genome- and protein sequences. A sequence dataset is sometimes referred to as a sequential or temporal dataset, in which case each data instance has an associated time-stamp. Another example of a sequential dataset is a transaction log from a sales system. A special case of a sequential dataset is the time series dataset where each instance is a sequence of measurements. Examples of time series data are stock data where the price of a stock varies over time or temperature data where the temperature varies over a period of time. In a spatial dataset, the data instances often include spatial attributes such as areas or positions together with other types of attributes. Data instances can be related through their spatial attributes, e.g., temperature measurements from two points on Earth close to each other. The spatial datasets can also include a time-stamp attribute and are then called spatio-temporal datasets. An example of this type of dataset is kinematic track data from a radar system where a physical object's position, speed and course are repeatedly measured. Graph datasets are another type of dataset where data instances or attributes of a data instance are related to each other through a graph structure. Examples of this kind of dataset are hypertext documents where web-pages are linked to each other and social networks where people are connected to each other based on social relations.

2.2.3.2 Type of Anomaly

The type of anomaly is another important issue to consider when dealing with anomaly detection. Chandola et al. [40] classify anomalies into three distinct categories: *point anomalies*, *contextual anomalies* and *collective anomalies*.

Point anomalies This is the most common type of anomaly and has been the focus in most of the anomaly detection research. A point anomaly is defined as a single data instance that is anomalous compared to the rest of the data instances [40]. In figure 2.7, the data instances a_1, a_2 and a_3 can be considered point anomalies. For example, consider a dataset with temperature readings from an engine, if the temperature value from one reading is outside the normal temperature range, the reading can be classified as a point anomaly.

Contextual anomalies A contextual anomaly is defined as a data instance that is anomalous with respect to the context. This type of anomaly is also referred to as a *conditional* anomaly [123]. Chandola et al. [40] state: "The notion of a context is induced by the structure in the dataset and has to be specified as a part of the problem formulation" and that the context can be defined by using *context-* and *behavioural* attributes. The context attributes for a data instance are used to define the context. In spatial datasets, the latitude and longitude of a location are examples of context attributes, while in a time-series dataset the time is the context attribute that describes the position of the instance relative to all other instances in the series. Behavioural attributes represent the non-contextual information in a data instance. For example, when measuring the temperature in a number of different places, the temperature value is a behavioural attribute and the latitude and longitude are the context attributes. If context attributes are available and context is an important property of the domain, context-based anomaly detection techniques should be considered [40].

It should be noted that, the definition of contextual attributes by Chandola et al. [40] differs from Steinberg's definition [125], presented in Chapter 1, in such a way that attributes (e.g. latitude, longitude, course) concerning a specific object are not regarded as contextual attributes.

Collective anomalies Chandola et al. [40] define a collective anomaly as a set of related data instances that are anomalous compared to the complete dataset. Each individual data instance need not be anomalous in itself, but the presence of multiple instances appearing together forms the collective anomaly. For example, it might be normal for a car to stop occasionally at a road intersection for a minute or two (perhaps waiting for a green light), but if the car stops multiple times within a short period of time, it could be a collective anomaly. The dataset for this event might have a number of consecutive data instances with a velocity attribute set to zero. It could be completely normal to

have a few such data instances, but anomalous if they appear in greater numbers. Chandola et al. [40] state that collective anomalies can only occur when the data instances are related. If contextual attributes are available, point and collective anomaly detection problems can also be transformed into a contextual anomaly detection problem.

2.2.3.3 Class Labels

The availability of class labels for data instances is essential for the choice of anomaly detection technique. When dealing with anomaly detection, the data instances can belong to one of the two classes *normal* or *anomaly* [40]. There are, however, problems related to the availability of class labels. When the datasets are recorded from a real-world system, the data instances are most likely unlabelled. The datasets must therefore be manually labelled by a human expert, which can be very costly when dealing with large datasets [40, 90]. According to Chandola et al. [40], it can be hard to obtain a representative dataset with data instances belonging to both the normal and the anomalous class. Anomalous data instances are dynamic in their nature and new types of anomalies can occur, for which no labelled training data exists. In some domains, the anomalies will correspond to rare events, i.e., two maritime vessels that collide and sink, for which no training data exists.

Anomaly detection techniques can be classified, according to the availability of class labels, into one of the following classes: *supervised anomaly detection*, *unsupervised anomaly detection* and *semi-supervised anomaly detection*.

Supervised anomaly detection When class labels are available, data mining methods for building predictive models can be used. There are, however, two problems with using predictive models for anomaly detection [40]. First, the number of data instances labelled normal vastly outnumbers the data instances labelled anomalous. This corresponds to the class imbalance problem that has been studied in the data mining and machine learning community; see Tan et al. [132] section 5.7 for more information about the class imbalance problem. Second, due to the fact that data instances from the anomaly class are rare, it could be hard to obtain a set of data instances that is representative for the class [40].

Unsupervised anomaly detection When using data from real-world systems, the class labels are often not available [132]. To be able to cope with this problem, a new type of anomaly detection called unsupervised anomaly detection was proposed by Portnoy et al. [105]. Unsupervised anomaly detection is based on two assumptions [105]:

1. "The number of normal instances vastly outnumbers the numbers of intrusions" (p. 2).

2. "The intrusions themselves are qualitatively different from the normal instances" (p. 2).

Unsupervised anomaly detection assumes that the training data probably includes anomalies. This should not affect the performance too much, as long as the anomalies are rare. The size of the percentage of anomalies that can be allowed depends on the context as well as on the specific normalcy modelling algorithm.

The second assumption states that anomalies should be distinguishable from the normal data, i.e., if the anomalies and the normal data look exactly the same, the anomalies will be impossible to find. How much the anomalies must differ depends on the context and the normalcy modelling algorithm. According to Tan et al. [132], a problem with this approach is that if many anomalies are similar to each other they might not be regarded as anomalies.

Semi-supervised anomaly detection In semi-supervised anomaly detection, both labelled and unlabelled data is used [132]. If the training dataset only includes data instances with the class label normal, a one-class classifier can be built. Since semi-supervised techniques do not require labelled data instances from the anomaly class, they are more applicable than supervised techniques [40]. The test data is classified as normal or anomalous depending on how well it fits the model. An example of a problem where semi-supervised anomaly detection can be used is failure monitoring systems. In such systems, data instances labelled normal can easily be recorded during normal operation, while anomalous instances can only be recorded during a failure. These failures can be too expensive to test, e.g., a spacecraft crashing, semi-supervised techniques must therefore be used. According to Chandola et al. [40], there are also some techniques that only use data instances labelled anomalous to build the model. The main problem with these techniques is obtaining a representative training dataset is very difficult.

2.2.3.4 Output from an Anomaly Detection Technique

The output from an anomaly detection technique is often used by an analyst for making decisions. It is therefore important that the output comes in a form that is suitable for the analyst. According to Chandola et al. [40], there are two common types of output from an anomaly detection technique, namely *scores* and *labels*. Some techniques can output a score that describes the degree of anomaly for a data instance. The score for each data instance can be presented to the analyst as a ranked list with the most anomalous instances at the top. The score can also be used together with a domain specific threshold to assign a label, such as normal or anomaly, to each data instance. Another class of techniques only outputs a binary classification, anomaly or not, for each data instance. In such cases, the analyst cannot directly affect the classification with a

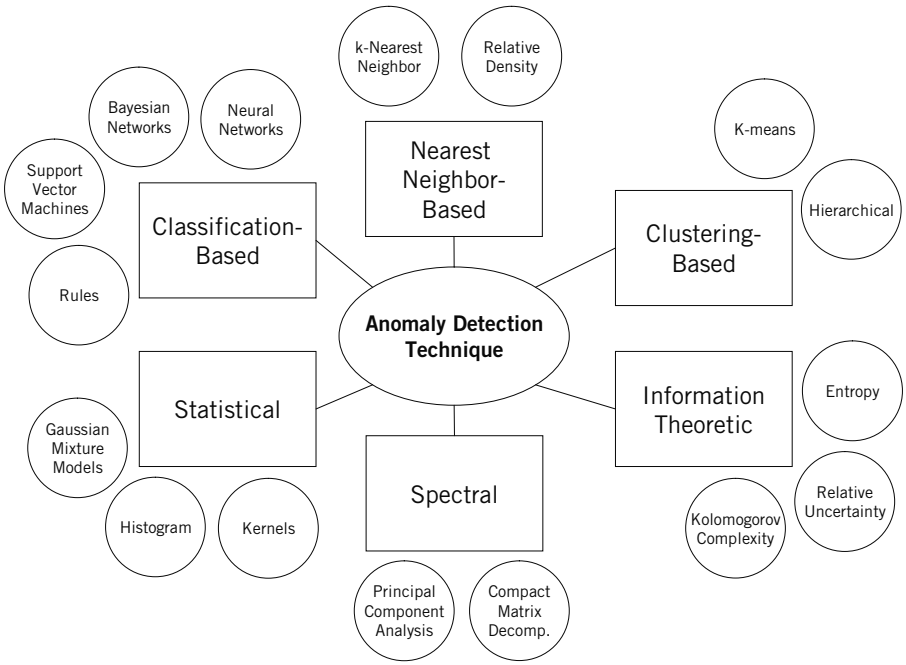


Figure 2.9: Anomaly detection technique classification, adapted from Chandola et al. [40].

threshold setting, but indirectly by setting specific parameters for the technique used.

2.2.4 Anomaly Detection Techniques

In this section, the main approaches for anomaly detection are presented together with their corresponding strengths and weaknesses. Figure 2.9 depicts a classification of anomaly detection techniques. The main classes of anomaly detection techniques are classification-based, nearest neighbour-based, clustering based, statistical, spectral and information theoretic. Each class of techniques has its pros and cons.

2.2.4.1 Classification-based Techniques

According to Tan et al. [132], classification is "the task of assigning object to one of several predefined categories". This is usually done by learning a classification function f that maps each attribute set a to a predefined class label b. Classification techniques can be used in two ways, to build *descriptive* mo-

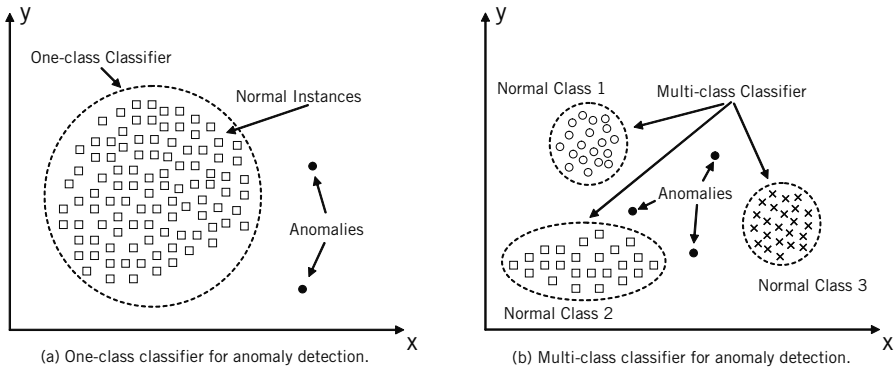


Figure 2.10: Two approaches for classification-based anomaly detection. Figure adapted from Chandola et al. [40].

dels and to build *predictive* models [132]. A descriptive model can be used to explain the differences between data instances belonging to different classes, while the predictive models can assign a class label to a data instance based on the attributes. In classification-based anomaly detection techniques, predictive models are built from labelled training data. These models are then used to classify new data instances as belonging to the class *normal* or the class *anomaly* [40]. According to Chandola et al. [40], there are classification-based anomaly detection techniques that can be regarded as *one-class* and *multiple-class* classifiers. Figure 2.10 shows examples of both types. The main difference is that for one-class classifiers, all the training data instances are assumed to be labelled normal while for multi-class classifiers the data instances in the training data can have a number of different normal labels.

Barbará et al. [17] use association rules [11] to generate a number of classes describing normal data instances. A data instance is considered to be anomalous if it cannot be classified as normal by any of the classifiers. Artificial neural networks have been used for building both single- and multi-class classifiers. Stefano et al. [124] trained a neural network on normal data instances with multiple normal classes. They then evaluated new data instances by feeding them into the network. If the network classified the instance as a member of a normal class, the instance was considered to be normal, otherwise it was considered to be an anomaly. Another technique often used for multi-class classification is according to Chandola et al. [40], Bayesian networks [73]. When dealing with univariate, conditionally independent data, the naïve Bayes classifier can be used [132]. If the attributes in the datasets are conditionally dependent, more complex Bayesian networks can be employed. Das and Schneider [44] proposed such a method based on the combination of multiple attributes in

a Bayesian network. Support Vector Machines (SVMs) [37] is another technique that can be used as a one-class classifier. SVMs can learn a region that encapsulates all normal data instances in the training dataset. In the test step, new data instances are classified as normal if they fall inside the learned region, otherwise they are classified as anomalous [117]. Rule-based anomaly detection techniques are commonly used for many applications and have been applied in both one-class and multi-class settings [40]. As in many of the previously described techniques, rule-based techniques work in two steps. In the first step, a rule learning algorithm such as RIPPER [42] or Decision Trees [62], is used to generate a number of rules describing the training dataset. In the next step, new data instances are evaluated by each rule to find the rule that best captures the data instance. Each rule has a corresponding support and confidence value. The support value is based on the number of data instances in the training dataset that match the rule divided by the total number of data instances. The confidence value is based on the number of data instances that match the rule divided by the number of instances covered by the rule. The confidence value can be inverted and used as an anomaly score for each rule. If the best matching rule has a very low confidence score, the data instance is considered anomalous [40]. Classification-based techniques often require labelled data.

The computational complexity of classification-based techniques is according to Chandola et al. [40] closely linked to the nature of the technique. In general, techniques based on decision trees are faster than techniques based on SVM or artificial neural networks. In the classification phase, all classification-based techniques are very efficient [40].

2.2.4.2 Nearest Neighbour-based Techniques

Nearest neighbour-based techniques are based on the assumption that normal data instances occur in dense neighbourhoods, while anomalous data instances occur in low density areas far from other data instances [40]. A central concept in nearest neighbour-based techniques is the distance or similarity measure [132]. The distance measure describes how close two objects are to each other. The closer the objects, the more alike they are. According to Tan et al. [132], it is very important to choose a distance measure that is appropriate for the problem and nature of the dataset. For datasets with continuous attributes, it is common to use distance measures such as the *Euclidian distance*, *city block distance* or the *Mahalanobis distance*, see [132] chapter 2 for a comprehensive introduction to distance measures. For categorical datasets, it is common to measure distance by just using a simple matching co-efficient, i.e., how many attributes with the same categorical value divided by the total number of attributes [23]. See Boriah et al. [23] for more information on distance measures for categorical attributes.

The Euclidian distance and city block distance are specific instances of the Minkowski distance measure (see Eq. 2.1)[132]. When r is set to 1, the Min-

kowski distance is equivalent to the city block distance and, when r is set to 2, the Minkowski distance becomes the Euclidian distance, n denotes the number of dimensions in the data instances:

$$d(x, y) = \left(\sum_{k=1}^{n} |x_k - y_k|^r \right)^{1/r} . \tag{2.1}$$

According to Tan et al. [132], the Euclidian distance measure is commonly used, but associated with a number of problems. First, if the attributes have different scales, the Euclidian distance measure will favour some of the attributes. Consider the task of calculating the similarity of people on the basis of their age and yearly income. For two given people, the difference in absolute income is likely much higher than the difference in age. The difference in income might be thousands of dollars while, the difference in age is at most 122[2], probably much less. Using the Euclidian distance measure, the difference in range between the age and income values is not regarded [132] and the income attribute will dominate in the distance calculation. The effects of this problem can be minimized by normalizing all attributes. Another problem with the Euclidian distance measure is that it does not handle correlations between the attributes [132]. To cope with this problem, the *Mahalanobis distance* can be used [97]. The Mahalanobis distance uses the variance in each attribute to calculate a weight for that attribute. The Mahalanobis distance between two vectors x and y is defined as:

$$mahalanobis(x, y) = (x - y) \Sigma^{-1} (x - y)^T , \tag{2.2}$$

where Σ^{-1} is the inverse covariance matrix. The Mahalanobis distance is a generalization of the Euclidian distance and can be used when the ranges of the attributes have different variances, the attributes are correlated, and the distribution of the complete dataset is approximately Gaussian [132]. According to Tan et al. [132], the Mahalanobis distance is more computationally expensive to calculate compared to the Euclidian distance.

There are two main classes of nearest neighbour-based anomaly detection techniques [40]. The first class uses the distance between a data instance and its k^{th} nearest neighbour as a degree of anomaly. The second class uses the relative density of each data instance to calculate the degree of anomaly. The relative density is calculated by measuring the distance from a data instance to its k^{th}nearest neighbours. This distance is the same as the radius of a hypersphere incapsulating the k nearest neighbours. The relative density can then be calculated by dividing the volume of the hypersphere with k. Nearest neighbour-based anomaly detection techniques do not require labelled data instances [40].

[2]According to Wikipedia, the French Jeanne Calment has the longest confirmed lifespan. Jeanne Calment lived for 122 years and 164 days.

2.2.4.3 Clustering-based Techniques

The goal of cluster analysis is to find groups of data instances that are closely related [132]. This differs from anomaly detection where the goal is to find data instances that are not closely related to the rest of the data instances. Even though there is a difference in the goals, cluster analysis can be used for anomaly detection. Clustering-based anomaly detection techniques can be ordered into three groups. In the first group, the assumption is that normal data instances belong to a cluster, while anomalous data instances end up outside all clusters [40]. Clustering techniques such as DBSCAN [57], ROCK [67] and SNN [55] have previously been used for this kind of anomaly detection. The second group of techniques is based on the assumption that normal data instances end up close to the centroid of the nearest cluster, while anomalous data instances end up far away from the centroid of the nearest cluster [40]. Anomaly detection based on this group of techniques is performed in two steps. First, the data instances from the training dataset are clustered. Second, the distance from each data instance, in the test dataset, to the closest cluster is calculated and used as an anomaly score. Popular clustering techniques based on this assumption are Self-Organizing Maps (SOM) [84], K-means Clustering [25] and Expectation Maximization (EM) [47]. In the third group of techniques, the assumption is that normal data instances are clustered into large and dense clusters, while anomalous data instances are either clustered into small or sparse clusters [40]. He et al. [72] proposed a technique called FindCBLOF that satisfies the assumption above. The method assigns an anomaly score called Cluster-Based Local Outlier Factor (CBLOF) to each data instance. The score is based on both the distance between the data instance and its nearest cluster and the size of the nearest cluster.

Clustering-based anomaly detection techniques share properties with the nearest neighbour-based techniques: both rely on a distance measure and their performance is dependent on how well the distance measure matches the properties of the dataset. According to Chandola et al. [40], a key difference between the two approaches is that clustering-based techniques use the closest cluster as reference, while nearest neighbour-based techniques use the local neighbourhood as reference when evaluating new data instances. Clustering-based techniques do not require labelled data, but can be computationally expensive. According to Chandola et al. [40], the computational complexity of building clusters is $O(n^2)$ for clustering techniques that require calculation of pairwise distances between data instances, where n is the number of data instances in the dataset. When the clusters are built, the complexity of testing new data instances is usually low, since the test data instance is only compared to a limited number of clusters.

2.2.4.4 Statistical Techniques

Statistical techniques use a statistical model to classify data instances. The model is built to reflect the distribution of the training data and new instances are classified on the basis of how well they fit the model. A data instance that is generated from the same stochastic process as the training data will fit the model well, while data instances generated from a different process will not fit the model very well and will be regarded as anomalies [132]. According to Chandola et al. [40], there are two groups of statistical techniques: *parametric* and *nonparametric*. When using a parametric technique, the underlying distribution of the data must be known or assumed to be known. If the assumption about the distribution is incorrect, the performance of the technique will decrease. There are a number of common distributions, such as Gaussian, Poisson and binomial, which can be applied to many types of datasets [132]. Examples of parametric techniques are Gaussian mixture models, regression models and techniques based on a mixture of parametric distributions. If the distribution of the data is unknown, nonparametric techniques can be used [40]. A nonparametric technique does not assume an a priori model. Instead, the model is built from the data instances in the training dataset. An example of nonparametric techniques is histograms, where the frequency of data instances in each bin are used to classify new data instances. Another example is kernel-function-based techniques where a kernel function is used to approximate the distribution of the data. The computational complexity of statistical techniques is dependent on the nature of the statistical model. Chandola et al. [40] made the following statements about the computational complexity of statistical techniques. Parameters for univariate models can be approximated in linear time, with respect to the number of data instances in the dataset. Fitting a multivariate model to a dataset usually requires the use of a technique such as Expectation-Maximization, which has a linear complexity (with respect to the number of data instances in the dataset) for each iteration. The number of iterations cannot, however, be approximated beforehand. Kernel-based techniques usually have a quadratic complexity. Statistical techniques do not require labelled data. When using histograms, the classification performance is, dependent on choosing suitable parameters for the histogram bins [40]. According to Chandola et al. [40], another problem with statistical techniques is that the assumptions about the distributions do not usually hold in high-dimensional real-world datasets.

2.2.4.5 Spectral Techniques

Spectral anomaly detection techniques are based on dimensionality reduction. It is assumed that the data instances can be transformed into a lower dimensional subspace in which anomalous instances can easily be distinguished from normal instances [10]. Parra et al. [102] evaluate techniques for anomaly detection based on three dimensionality reduction methods: *multidimensional scaling*, *lo-*

cally linear embedding and *isometric feature mapping*. Another commonly used technique for dimensionality reduction is *Principal Component Analysis* (PCA) [78]. In PCA, the number of dimensions in a dataset with correlated attributes is decreased while preserving as much of the variation existing in the original dataset as possible [50]. According to Kargupta et al. [80], the result of PCA is a number of statistically uncorrelated principal components, which are ordered on the basis of their variance. The principal components can be used to test how well new data instances fit the lower dimensional distributions of the original dataset. If the new data instances do not fit they can be regarded as anomalies [80]. Sun et al. [130] used a similar approach, when they proposed Compact Matrix Decomposition (CMD) to find anomalies in sequences of graphs. The computational complexity of PCA is typically linear to the number of data instances and quadratic to the number of attributes in each data instance [78]. Spectral Techniques do not require labelled data [40].

2.2.4.6 Information Theoretic Techniques

Information theoretic techniques use measures, such as entropy, relative entropy and Kolomogorov complexity to analyse the information content in a dataset [40]. The assumption is that anomalies "...induce irregularities in the information content..." [40]. Let $C(D)$ denote the complexity of a dataset D. The Pareto-optimal solution can then be found by searching for the minimal subset of instances, I, that result in a maximization of $C(D) - C(D - I)$ [40]. This requires a multi-objective optimization and, hence, the computational complexity is exponential, with respect to the number of data instances. Examples of complexity measures used for anomaly detection are the size of regular expressions [15] and the size of a data file compressed with a standard compression method [82]. Information Theoretic Techniques do not require labelled data and do not require any assumptions about the statistical distribution of data [40]. According to Chandola et al. [40], one problem with this kind of technique is that it is hard to apply on sequential and spatial datasets.

2.2.5 Signature Detection

In knowledge-based anomaly detection, the nature of the anomaly is known beforehand by eliciting information from a domain expert. This type of anomaly detection is also referred to as signature-, rule-, template- or case-based detection. A example is [66], where Guerriero et al. propose a system that searches for anomalies like "small boats on open sea" or "large ships without AIS transponder". These two anomalies are known in advance and can therefore be built into the detection system.

Fooladvandi et al. [59] proposed a signature detection system based on Bayesian networks acquired from expert knowledge to detect a maritime piloting scenario. The pilot scenario was divided into a number of sub-scenarios

representing specific signatures. The signatures were then fused together in a Bayesian network to decide if a piloting scenario occurred or not. The system was evaluated on real-world maritime AIS data and the evaluation showed that the approach could be used to detect real-world piloting scenarios.

Edlund et al. [51] presented a rule-based approach for situation assessment targeted at the domain of sea-surveillance. The system uses a specifically engineered ontology together with an agent-based architecture to reason about vessel behaviour in an area of interest. The system also has a graphical rule-editor for building complex rules based on concepts in the ontology.

Seibert et al. have used a hybrid approach for the SeeCoast system presented in [121]. The system extends the US Coast Guard Port Security and Monitoring system by adding automated support for detection, classification and tracking. The system can learn normalcy models from data for the detection of anomalous vessel behaviour. Another part of the system is the rule-based pattern recognition component which can detect pre-defined patterns of interest.

2.2.6 Hybrid Systems

Recent research has suggested that the performance of current anomaly detection systems can be improved, by taking a hybrid approach with both anomaly detection and signature detection [103]. The two approaches complement each other; anomaly detection helps in finding novel or unknown events without defining them in advance. In a new system, where there is no or just a small set of training data, anomaly detection methods will have inadequate performance, because there is not enough data to build a useful normal model. In that case, the signature detection will at least be able to find events that have been defined in advance by subject matter experts and hence improve the performance of the total system. Some events are very easy to find by just defining their signature, while it is almost impossible to define an adequate signature for other events.

According to Patcha and Park [103], a combination of knowledge-based and data-driven methods can best detect both known anomalies and new, previously unknown ones. This concept resembles Situation Management described in Section 2.1.5.

2.3 Uncertainty Management

Uncertainty management is a key concept in Information Fusion, which is mainly due to the nature of the information that is processed. The information often comes from physical sensors that measure some kind of phenomena in the real world. In these measurements, a number of uncertainties arise in processes such as signal processing, tracking, situation assessment and so on. The role of uncertainty management is to handle these uncertainties in a structured way. According to Berztiss [20], there are many methods for dealing with uncertainty, such as Bayesian estimation, fuzziness, time Petri nets, rough sets,

belief theory, evidence theory and possibility theory. In anomaly detection, uncertainty management plays an important role in minimizing the impact of uncertainties, when classifying objects as normal or anomalous.

2.3.1 Bayesian Theory

Assume that we are interested in a random variable, X, with a corresponding, discrete state space, Ω_X, and where observations y_1 and y_2 are considered to be *evidence* regarding the true state of X. For anomaly detection problems, the state space $\Omega_X = \{normal, anomaly\}$ is used. In Bayesian theory [18], such evidence is represented by *likelihood functions* $p(y_1 \mid X)$ and $p(y_2 \mid X)$, respectively. By assuming that the observations y_1 and y_2 are *conditionally independent* given X, we can construct the joint evidence $p(y_1, y_2 \mid X)$ by:

$$p(y_1, y_2 \mid X) = p(y_1 \mid X)\, p(y_2 \mid X). \tag{2.3}$$

One problem with Equation 2.3 is that the joint evidence monotonically decreases with the number of observations, e.g., multiplying two evidences, each with a value less than one, will result in a value lower than both of the multiplied evidences. Thus, when implementing a combination operator based on Equation 2.3, such an operator will eventually yield erroneous results due to the limited precision of the double representation. For this reason, it is convenient to formulate Equation 2.3 in an alternative way where the likelihood functions and the result of the combination have been normalized (cf [14, 13, 81]):

Definition 1: *The Bayesian combination operator* $\Phi_{\mathcal{B}}(\hat{p}(y_1 \mid X), \hat{p}(y_2 \mid X))$ *is defined as:*

$$\Phi_{\mathcal{B}}(\hat{p}(y_1 \mid X), \hat{p}(y_2 \mid X)) \triangleq \frac{\hat{p}(y_1 \mid X)\, \hat{p}(y_2 \mid X)}{\sum_{x \in \Omega_X} \hat{p}(y_1 \mid x)\, \hat{p}(y_2 \mid x)}, \tag{2.4}$$

where $\hat{p}(y_i \mid X)$, $i \in \{1, 2\}$, *are conditionally independent normalized likelihood functions. The operator is undefined when* $\sum_{x \in \Omega_X} \hat{p}(y_1 \mid x)\, \hat{p}(y_2 \mid x) = 0$.

The Bayesian combination operator is used in Chapter 6.

Chapter 3
Anomaly Detection in the Surveillance Domain

This chapter addresses research objectives one and two. The main contributions of the chapter are:

- A characterisation of the surveillance domain.
- A list of important properties for evaluating anomaly detection methods.
- A review of existing anomaly detection methods used in the surveillance domain.

3.1 Properties of the Surveillance Domain Related to Anomaly Detection

The surveillance domain, which is the intended scenario domain of this research, is characterised by an area of interest that is important for an analyst to know about in order to be able to find threats or other interesting activities. A large number of objects (hundreds or possibly thousands [110]) are present in the area. The area of interest is surveyed by a number of sensors and sometimes also directly by humans, e.g., in the maritime domain, vessels are surveied by both radar sensors and AIS (Automatic Identification System), as well as visually and via radio communication. The sensors generate a constant flow of kinematic data about objects in the area of interest, e.g., position, speed and course [110]. As a complement to the kinematic data, there is also contextual information about the domain (e.g., maps, time tables and weather information) which can be used to constrain the processing [125] and expert knowledge about what can be expected in the area. The expert knowledge can for example consist of thresholds and parameter settings used when building normal models and detection anomalies but also which attributes in the input data to use and

how the pre-processing is done. If humans survey the area, their observations are often reported in a non structural format (e.g. "I observed an activity occurring near the Market square involving a small group of men...") [70]. The data from the sensors is unlabelled, i.e., the sensors do not have enough information to put a label (normal or anomalous) on the observed objects. Due to the volume of data flowing into a surveillance system, the operators cannot be expected to label incoming data [110].

A central concept in the surveillance domain is a *track*. In this work, a track is defined as: *a number of consecutive observations of an object ordered by the time of the observation.* An observation is defined as: *a measurement of the kinematical properties of a physical entity, at a specific time, by a sensor*. Kinematical properties are defined as: *the information about an object's kinematical state, e.g., position, speed and course, related to the object itself*. An observation is equal to the concept of a *track report*, used by Rhodes et al. [110], and is the result of one iteration in the JDL level 1 data association process (e.g. a Kalman or Particle filter) [93], p. 72. Hall et al. [70] classifiy sensors as hard and soft. A hard sensor is another term for the traditional electronic sensor, while a human delivering a report can be considered a soft sensor.

In the surveillance domain, data from the sensors has a low number of attributes (e.g. latitude, longitude, id, time-stamp, speed and course), at least compared to the network intrusion domain, where the datasets can have a large number of attributes [90]. For example, the KDD Cup 1999 intrusion detection dataset [1] includes 42 attributes. Datasets collected from sensors can be very large; video trackers can update hundreds of tracks (with six to ten attributes), fifteen times per second, which results in about 130 million updates each day [30]. Another important property of the surveillance domain is that data must be processed in real-time. This constrains the processing time of the parts of the system responsible for analysing data. The handling of temporal aspects is another difference between the surveillance and the network intrusion domain. In the network intrusion domain, the temporal aspects are usually handled by just adding an attribute; an example of such an attribute is *NumberOfConnectionsFromSourceLastFiveSeconds* [89]. In the surveillance domain, temporal aspects are handled by multiple observations of the same object at different points in time.

There are a number of different application areas within the surveillance domain, e.g., land transport, air, under water, maritime and public areas. Some of these areas will be addressed in this thesis.

3.2 Evaluating Anomaly Detection Methods Used in the Surveillance Domain

When evaluating anomaly detection methods, both quantitative (e.g. precision and recall) and qualitative (e.g. robustness and adaptability) aspects must be

considered. The quantitative aspects can be measured by conducting empirical experiments, while the qualitative aspects are subjective and hard to measure. However, there are qualitative properties that are easy to measure, e.g., whether the model is a black-box or white-box model.

3.2.1 Quantitative Evaluation

The performance of an anomaly detection method can be quantitatively evaluated by measuring computational cost, precision and recall. The methods also have a number of important properties that can be qualitatively evaluated. Many surveillance systems are dependent on input data from sensors, which often deliver data into the system within fixed context-dependent intervals. For example, a video camera might generate 25 frames per second. If the method cannot meet the deadlines, it could have a negative impact on the performance or not deliver any result at all. It is also important to measure the computational cost when building the normal model. If a method has low computational cost when building the normal model, it is much easier to rebuild the normal model when the behaviour of the objects in the domain changes and the system downtime can be reduced.

Precision and recall measure the exactness and completeness of the method. In the anomaly detection scenario, the precision is the ratio between the number of anomalous objects classified as anomalies and the total number of objects classified as anomalies. A high precision is desirable, because that minimizes the number of normal objects wrongly classified as anomalies. These are often called false positives or false alarms. However, it is not certain that these objects are really uninteresting to an operator. The recall value represents the ratio between the number of true anomalous objects classified as anomalies and the total number of anomalous objects in the dataset, i.e., how many of the anomalous objects are found. For example, a precision of 95% indicates that 95 out of 100 objects classified as anomalies by the method are, in fact, anomalous, while 5 out of 100 are incorrectly classified as anomalous by the system. A recall of 70% is the result of a method that finds 7 out of 10 objects labelled anomalous.

3.2.2 Qualitative Evaluation

Based on the aim of this work, the following properties have been identified as important with respect to the use of anomaly detection in the surveillance domain:

- Handling of contextual information

  Contextual information is very important in Information Fusion as well as in many surveillance domains [125, 61]. Therefore, the anomaly detection method should be able to take advantage of contextual informa-

tion and use it together with kinematic information [110]. This property can be assessed by analysing whether the representation and normalcy modelling of an anomaly detection method, can incorporate contextual information.

- Transparency

  According to Setnes et al. [122], transparency in the area of data mining is, a measure of how easy it is for a human to interpret and analyse a model. Models are usually grouped into two classes; 1) *white-box* models, with a high degree of transparency, e.g., decision trees, and 2) *black-box* models, with a low degree of transparency, e.g., artificial neural networks (ANN). In anomaly detection, the transparency of the normal model can be of great importance for the human operator. First, a more transparent model enables the possibility of explaining what caused the anomaly. In a decision tree, the route from the leaf node to the root node can be used to explain the cause of a classification. Compare this to an ANN, where the internal weights say little about what caused the classification. If a model is transparent to the user, the users trust in the system could increase, due to the fact that they easier can understand why the system made a certain classification. This property will be judged to be fulfilled if the evaluated approach, is based on a method that is using a white-box model or a model that can be visualised to the operator, otherwise the property will be assessed as unfulfilled.

- Incorporation of expert knowledge

  The operators of a surveillance system often have in depth knowledge about what can be expected in an area of interest [110]. If this knowledge can be captured in the normal models, the performance of the system should increase, which might also increase the operators' trust in the system. This property is judged to be fulfilled, if the evaluated method allows for the inclusion of expert knowledge.

- Robustness

  Robustness is defined by the IEEE as "*the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environment conditions*" [2]. Related to anomaly detection, the robustness measure used in this work is based on the following factors: (1) How well the model handles incorrect data or noise?, c.f., Witten and Frank [137], p. 313. This factor is exemplified by anomalies in the training data, which should have a minimal impact on the overall performance and would not cause the model to break down. (2) How well the model handles both small anomalies (with a minor deviation from normality) that develop over time as well as large ones? This property will

be judged on the basis of whether the evaluated method handles the two factors decribed above.

- Adaptability

  The adaptability is a measure of how easy it is to change definitions of normality. According to Rhodes et al. [110], this is a necessity in an operative system due to the fact that the world is in constant change. Consequently, the notions of normal and anomaly also change over time. Another aspect of adaptability is a normal model that constantly grows when new data comes into the system. This property will be judged to be fulfilled, if the evaluated method allows for the normal models to be updated at run-time, without being reconstructed from scratch, i.e., if the models can be incrementally updated.

- Handling of joint attributes

  In the surveillance domain, kinematic features are often joint ones, e.g., the latitude and longitude positions must be used together to be of any use. It does not make sense to model the latitude and longitude attributes separately, the meaning of such a model is hard to imagine. The same is true for the speed and course features, which should be used together with the position. It is of course possible to create a normal model with just position and speed and another with position and course. However, none of these models captures the aspect that it can be normal to travel fast and north at a certain position, but not to travel slowly and north. To be able to capture all these aspects in a normal model, it must be able to handle joint attributes. This property is identified by Tan et al. [132] as an important anomaly detection issue. The property will be judged on the basis of whether the evaluated method takes joint attributes into account or not.

3.3 Previous Methods for Anomaly Detection in the Surveillance Domain

This section reviews a number of papers that describes different methods used for anomaly detection, in the surveillance domain. In this research, the focus is on ground and maritime surviellance, therefore, the choise of papers has been restricted to papers from conferences and journals addressing with these areas. The reviewed papers were published in the proceedings of conferences and journals on Information Fusion, Sensor and Signal processing, Video Image Analysis and Visual Surveillance. The papers have previously been used and analysed in the publications on the SBAD method. Some of the papers in the review, referes to other papers, also in the review. In this chapter a structured analysis of the reviewed papers is presented. Each method is evaluated on a specific, often

proprietary, dataset. Therefore, the accuracy of each method cannot be directly compared on the basis of the results presented in the papers. Consequently, the methods are only subjected to a qualitative evaluation based on the properties in Section 3.2.2.

3.3.1 Automated Anomaly Detection Processor

Kraiman et al. [85] propose a system called Automated Anomaly Detection Processor (AADP). AADP can find events of interest in tracking and surveillance data and has been used in several civilian and military applications. The first step in the AADP algorithm is to select a number of attributes that are used to describe the behaviour of objects of interest. The attributes can include both "raw" attributes that comes directly from a sensor and those that have been pre-processed. Each observation of an object is represented by an N-dimensional vector. A sequence of such vectors describes the behaviour of the object over time. A number of such sequences with recorded data form the training dataset. The next step is to apply a clustering algorithm to the training dataset, in the case of AADP, Self-organizing maps (SOM) [84] are used for clustering. The SOM algorithm clusters all N-dimensional vectors into a number of model vectors that represent the typical or "normal" behaviour of the objects in the dataset. This is done without inserting any *a priori* knowledge about normal behaviour. The SOM model can be used to classify new observations as normal or anomalous, but does not provide automated anomaly detection. This is because many observations do not fall into well-defined clusters. Using only the SOM model also makes it hard to regulate the false alarm rate of the system, i.e., it is difficult to set a threshold that will result in a fixed number of alarms. To automate the anomaly detection, AADP uses Gaussian Mixture Models and Bayesian analysis. Each node in the GMM is represented by an N-dimensional probability density function (PDF). The means in the GMM are set from the SOM model vectors and the variances are set from the dispersion of the training data instances around each model vector. A weighted average of each node in the GMM is used to calculate the probability of a new observation behaving normally. The GMM can also be used to characterise which attributes in a new observation deviate the most from the normal values of that attribute. The probability for normality of an observation is accumulated over time within a sliding window. This enables AADP to both detect large anomalies (which reach the detection threshold within minutes) as well as relatively small but consistent anomalies (which develops during tens of minutes before reaching the alarm threshold). The last step in the process is to alert the operator of the system.

At first glance, the AADP appears to be an attractive method for maritime anomaly detection. However, on closer look, the method has a number of drawbacks. First of all, the authors do not present any figures of what kind of performance can be expected in terms of true positive rate for a specific false

alarm rate. These kinds of performance figures can be hard to obtain, when dealing with real-world datasets, but all alarms can be analysed to see which of the reported anomalies an operator would classify as anomalous. Moreover, the description lacks the degree of detail that is needed to re-implement the method. The overall method is well described, but specific parameters and some of the design considerations are not. Another problem with the method is that the SOM needs a suitable distance function. This problem is discussed in Section 2.2.4.2. The dataset used for the evaluation is not publicly available; it is therefore impossible to make a direct comparison between AADP and other methods. The AADP handles context information based on the timestamp of an observation, there is no information about how to include other contextual attributes. Therefore, the contextual information property is judged to be fulfilled to some degree. The actual anomaly detection is done by using a multi-dimensional GMM, which is hard to visualise if it includes more than two dimensions. Therefore, the transparency property is judged to be unfulfilled. By allowing the operator to set thresholds and pre-processing options, the expert knowledge property is judged to be fulfilled. The ability to handle both large and small anomalies leads to the assessment that the robustness property is fulfilled. There is no information in the paper whether the normal models can be continuously updated. In principle, it should be possible to update the SOMs without rebuilding them from scratch, therefore the adaptability property is fulfilled to some degree. A problem with SOMs is that the initial model vectors can have a great impact on the end result. It is not clear how this is handled in AADP. Joint attributes are handled in the SOM, but after the clustering is accomplished, some of the information about correlations is lost. Therefore, AADP is judged to fulfill the joint attributes property to some degree.

3.3.2 SeeCoast Port Surveillance

A method for maritime situation monitoring and awareness has been proposed by Rhodes et al. [108]. The method uses a cognitively inspired algorithm to learn the normal behaviour of vessels. More precisely, the learning algorithm is based on “a significantly modified version of the Fuzzy ARTMAP neural network classifier” - Seibert et al. [121]. The algorithm implements an unsupervised clustering of feature vectors into a number of clusters, with each cluster representing normal behaviour. A user-specified threshold, called *vigilance*, is used to adjust the specificity of the learned model. The approach is evaluated on both artificial data, as well as on 20 hours of real-world data recorded around New York harbour. In the paper, Rhodes et al. demonstrate the method’s ability to learn normal behaviour and, with the help of an operator, improve the normal model, as new data arrives into the system. The proposed method has a number of compelling features (see Table 3.1), but there is unfortunately no information about the representation except for simple examples with speed

connected to different areas. Moreover, there is no information about which modifications have been done to the original Fuzzy ARTMAP approach.

Improvements to the method were presented by Bomberger et al. [22] and Rhodes et al. [109]. The first was the ability to predict future vessel behaviour, another improvement was the representation of features. In the original approach, only speed and position were used, but in the improved version, a four-dimensional feature vector, including latitude, longitude, speed and course, was used. The course feature was discretised into four states (north, east, south and west) and the speed feature was discretised into three states (slow, medium and fast). These two features were combined into a total of 12 different states plus one called *stopped*, which was used when the speed was below a threshold. The longitude and latitude attributes were discretised into a uniform grid. The results showed that it was possible to predict future vessel locations based on current vessel behaviour. No more details about the modified Fuzzy ARTMAP classifier were presented in the paper. In Rhodes et al. [109], the future vessel location prediction was improved by using multiple spatial scales. A complete system called SeeCoast port surveillance is presented in [121]. The method can include contextual information, such as time of day and geographic information. Therefore, it is judged that the method fulfills the contextual information property. The Fuzzy ARTMAP neural network classifier is a black box model, but Rhodes et al. [110] show that it is possible to visualise some parts of the normal model, therefore, the approach is judged to fulfill the transparency property to some degree. The method can include expert knowledge and update the normal model continuously, and, therefore, the expert knowledge and adaptability properties are judged to be fulfilled. In addition, the method supports joint attributes of kinematic features, and is therefore judged to fulfill the joint attributes proptery to some degree. The robustness property is judged to be fulfilled on the basis of the use of a multi-scale model in which multiple models with different time scales are built and the most appropriate model, based on a number of parameters, is used for classification.

3.3.3 Momentary Track Features Modelled by Gaussian Mixture Models

Laxhammar [87] proposed a method for detecting anomalous vessel behaviour based on unsupervised clustering of information from normal vessel behaviours. The method uses four-dimensional Gaussian Mixture Models (GMMs) to capture normal behaviour. The area of interest is divided into a grid where the behaviour of vessels in each grid cell is modelled as a separate GMM. The GMMs are built by using a greedy version of the Expectation-Maximization algorithm. The method was evaluated offline on a recorded real-world dataset containing sea traffic from south of Sweden. The result of the evaluation shows that the proposed method can find anomalies, such as vessels crossing sea lanes and those travelling in the opposite direction close to sea lanes. The

proposed method uses a statistical approach that does not require any distance metric. As presented in [87], the method does not include any mechanism for including contextual information. The four-dimensional GMM cannot easily be visualised with the standard techniques used for two-dimensional GMMs (see Figure 4.4 for a visualisation of a two-dimensional GMM). Therefore, the transparency property is judged to be unfulfilled. In addition, the method does not support inclusion of expert knowledge or the ability to adapt the normal models. The robustness property is judged to be fulfilled to some degree; the model can handle noise in the input data by lowering the detection threshold, but does not handle anomalies that develops over time. The method uses the four-dimensional GMM to support joint attributes.

3.3.4 BALDUR

Gutchess et al. [68] introduce a system called BALDUR (Behaviour Adaptive Learning during Urban Reconnaissance) that learns probabilistic models of observed activity for a given location. The system is based on unsupervised learning techniques and can both learn patterns of normal activity as well as find suspicious or unusual behaviour in real-time. Furhtermore, the system uses CCTV cameras and stores all information in a database. While the information in the database is mainly used for training the system, it can also be used for forensic analysis, in which the operator can ask the system questions like "Show all pedestrians that approached the entrance door in building X between time t_1 and t_2". The system was evaluated using a large dataset containing a month of observed activity from a commercial parking lot. In the evaluation, all object positions were discretised into a uniform grid with 3m by 3m cells. The anomaly detection was based on hidden Markov models (HMMs) [106] that represented probabilities for normal transitions between cells in a rectangular grid. The result of the evaluation revealed that the system was able to find anomalies, such as cars arriving/departing at unusual hours and pedestrians running in the parking lot. The dataset that was used for the evaluation is not publicly available. Therefore, it is impossible to make a direct comparison between BALDUR and other methods. The description of the method also lacks the specific parameter settings used in the experiments. Although, the HMM approach with state transitions is interesting, the proposed method only handles positional information. The BALDUR system does not support the inclusion of expert knowledge, but uses a number of different models to capture temporal variation; therefore, contextual information is supported to some degree. Since it is possible to visualise the internal structure and probabilities of an HMM, the transparancy property is judged to be fulfilled. The system can detect anomalies that develop over time and the position discretisation makes the method less sensitive to noise in input data, the robustness property is therefore judged to be fulfilled. However, the normal model cannot be updated, and, therefore,

the adaptability property is judged to be unfulfilled. Joint attributes are handled to some degree, i.e., the x and y positions are joined together.

3.3.5 Kernel Density Estimation for Maritime Anomaly Detection

Another approach for the statistical analysis of motion patterns in AIS data was presented by Ristic et al. [111]. Their method is based on adaptive Kernel Density Estimation (KDE), also known as the Parzen Window method. The adaptive KDE approach can be used for estimating unknown probability densities using only a training dataset. In the adaptive version of KDE, the width of each kernel is parameterised and can be adjusted to reflect variations in the distribution of the training data. A four-dimensional state space, including latitude, longitude, speed and course, was used as input data for building the KDE model. The method was evaluated on both simulated and on real-world data recorded in Gulf St Vincent (Port Adelaide). Both the tasks of anomaly detection and that of motion prediction were evaluated, and the experiments showed that the method was able to find a number of artificial anomalies in both real-world and simulated data. Two different methods for motion prediction were presented and evaluated. These evaluations demonstrated that the prediction could be improved by indexing the training data based on the origins of vessels. The method uses contextual information to some degree, i.e., the use of the origin of vessels. It is hard to visualise a four-dimensional KDE model and, therefore, the transparency property is judged to be unfulfilled. Furthermore, the method does not allow for any way of including expert information, and if the adaptive version of KDE is used, the normal model cannot easily be modified. The robustness property is deemed to be fulfilled to some degree; the model can handle noise in the input data but does not seem to manage anomalies that develop over time. The approach supports modelling of joint attributes.

3.3.6 Activity Recognition and Abnormality Detection (ARAD)

Duong et al. [49] addressed the problem of learning and recognizing human activities. The domain is visual surveillance and the aim of the work is to find methods to automatically build models of normal activities and their duration, and to find anomalous activities based on these models. The activities are modelled using a two-layered extension of the hidden semi-Markov model. While the bottom layer is used to model atomic activities and their duration, the top layer models high-level activities that are built from a sequence of atomic activities. The experimental setup is a kitchen with a CCTV camera mounted in each corner, and a tracking system producing tracks of the person in the room. The positions are discretised into 1 m by 1 m cells. The high-level activities used in the experiments were eating-breakfast, washing-dishes, making-coffee and so forth. These activities were usually performed in a specific sequence.

The system was trained using a number of sequences with typical durations for each high-level activity. In the testing phase, Duong et al. were able to find both normal and anomalous activity sequences, as well as sequences where the duration of specific activities deviated from the normal model. The method only handles positional information and temporal sequences of transitions between different positional cells. Although, the approach is similar to the one used in the BALDER system, in ARAD, the movements of people is mapped to high-level behaviour. It should be possible to extend the method so that it can handle more attributes than just the position attribute. The method can find anomalies based on deviating state transitions as well as those where an object stays in a state too long. The method was evaluated on a small synthetic dataset that contained only six high-level activities. It is not clear how this approach will scale when the number of high-level activities increases. Since the method includes the duration of activities as contextual information, it is judged that the method fulfills the contextual information property to some degree. Some of the transition probabilities of the proposed model can be visualised. Hence; hence, the transparency property is fulfilled to some degree. However, the approach does not support the inclusion of expert knowledge or incremental updates. Joint attributes are handled to some degree, i.e., the x and y positions are joined together.

3.3.7 Hierarchical Hidden Markov Models for Activity Recognition (HHMM)

Bui et al. [36] used hierarchical HMMs to find anomalous behaviour among people at an airport, by analysing transitions between different areas of the airport and assigning them to a number of pre-defined behaviours. Three high-level behaviours (exit-airport, pickup-luggage-exit and meet-pickup) were defined using nine combinations of lower-level behaviours (exit-airplane, turn-left-foyer, turn-right-carousel, pass-left-right, pass-right-left, collect-luggage, leave-carousel, enter-foyer and leave-foyer). The HHMM uses a four-level hierarchical topology and the probabilities in the model are estimated using the Expectation-Maximization (EM) algorithm. The data used in the airport experiment included simulated movement trajectories. The results show that the proposed method requires less computation time compared to previous approaches. Another result is that the proposed method can learn to detect nine different high-level behaviours. If a behaviour does not fit the learned model, it can be regarded as an anomaly. The method does not support the inclusion of contextual information or expert knowledge. In principle, the learned model can be visualised by an operator. The authors do not present any information regarding the support of joint attributes. Since the method uses the EM algorithm [47] to build the normal model, it cannot be easily updated. The method can be considered to handle noise and detect behaviours that develop over time.

3.3.8 Qualitative Evaluation of Previous Anomaly Detection Methods

Table 3.1 shows the results of the qualitative evaluation of anomaly detection methods used in the surveillance domain, relevant to the application areas of maritime and ground surveillance. These assessments are derived from the description of each method and the algorithms on which they are based. However, based on the available information about the model, it was sometimes unclear whether a method fulfilled a certain property or not. The fulfillment of each property is assessed as: *no*, *to some degree* and *yes*, where *no* corresponds to a method that does not fulfill a property and *yes* to a one that completely fulfills the property. As the table indicates, all methods, except the SeeCoast one, handle contextual information poorly; the degree of transparency and usage of expert knowledge are also low for most methods. While all of the methods can be considered to be robust, there are adaptability problems for all of them except SeeCoast and AADP. In addition, there are some problems with handling joint attributes in the input data.

3.3.9 Distance Measures for Maritime Data

AADP and many other anomaly detection techniques use distance measures to calculate similarity between data instances. It is vital to select a distance measure that is suitable for the dataset, to avoid unintentional side effects [40]. An example of an unintentional side effect is the problem when some attributes are assigned a much higher weight compared to other attributes, due to different range scales (c.f. the age and income example in Section 2.2.4.2). Another problem is the mixing of different types of attributes (c.f. integral and separable attributes [65]). In this section, recorded real-world Automatic Identification System (AIS) data is analysed to evaluate whether it makes sense to use distance measures like the Euclidian or the Mahalanombis distance. This is related to the goal of anomaly detection, that is, is to aid a human analyst and it is important that the analyst can understand what caused a specific result.

3.3.9.1 Maritime Dataset

The maritime dataset used in the analysis consists of eight days of AIS data recoded on the Swedish west coast. The dataset contains 6,742,641 AIS reports from 976 unique vessels. This AIS data includes a number of attributes organised into two groups. The first group contains dynamic data, such as latitude and longitude position, speed over ground, course over ground, navigational status and so on. The second group contains static data, such as the dimensions of the vessel, ship type and so forth. A complete list of the attributes in the AIS data can be found in the AIS specification ITU-R M. 1371 [74]. Table 3.2 illustrates the attributes used in the analysis and their corresponding ranges. Note

Table 3.1: Results of the qualitative evaluation. Each property is assessed as "No" if the property is not fulfilled, "Yes" if the property is fulfilled completely and "To some degree" if the property is partially fulfilled.

	AADP	SeeCoast	GMM	BALDUR	KDE	ARAD	HHMM
Contextual information	To some degree	Yes	No	To some degree	To some degree	No	No
Transparency	No	To some degree	No	Yes	No	To some degree	Yes
Expert knowledge	Yes	Yes	No	No	No	No	No
Robustness	Yes	Yes	To some degree	Yes	To some degree	Yes	Yes
Adaptability	To some degree	Yes	No	No	No	No	No
Joint attributes	To some degree	To some degree	Yes	To some degree	Yes	To some degree	No

Table 3.2: AIS attributes used in distance measure analysis.

Name	Range	Unit	Resolution
Latitude	$[-90, 90]$	Decimal degrees	$\frac{1}{10000}$ degree
Longitude	$[-180, 180]$	Decimal degrees	$\frac{1}{10000}$ degree
Course	$[0, 359]$	Decimal degrees	$\frac{1}{10}$ degree
Speed	$[0, 102.2]$	Knots	$\frac{1}{10}$ knot

that according to the AIS specification, a speed of 102.2 indicates 102.2 knots or higher.

3.3.9.2 Euclidian Distance

The Euclidian distance is one of the most commonly used measures for similarity [40]. This measure tends to favour attributes with large ranges over attributes with small ranges (compare the age and income example in Section 2.2.4.2). As Table 3.2 indicates, the attributes in the AIS dataset have ranges in the same magnitude, but there are still some differences in range. Consider the latitude and longitude attributes, where the range of the longitude attribute is twice as large as the range for the latitude attribute. This might lead to longitude domination on longitude values greater or less than 90 degrees. A standard way of dealing with this problem is to normalize the attributes between zero and one, which raises another problem; the latitude and longitude are not linearly correlated. At the equator, the distance in meters of one latitude degree and one longitude degree is the same. However, when the latitude increases (going north from the equator) or decreases (going south from the equator), the distance in meters of one longitude degree will decrease due to the Earth's spherical shape. This will favour the latitude coordinate in the Euclidian distance calculation. Although the latitude and longitude are somewhat different, the difference between the other attributes is much greater. Consider using the longitude and course attributes in a Euclidian distance calculation. They both have a range of 360 degrees, but can they be directly compared? If the longitude attribute is changed by 1.0 degree, it does not mean the same as when the course attribute changes by 1.0 degree. A change of 1.0 degree on the longitude attribute equals up to 111 km (at the equator). To a human, this change is much greater than the change of course by 1.0 degree. Using these two attributes is like comparing apples with pears, it can be done mathematically, but makes little sense. The result of such a comparison can be used in clustering-based techniques but it will form clusters that potentially make no sense to a human analyst.

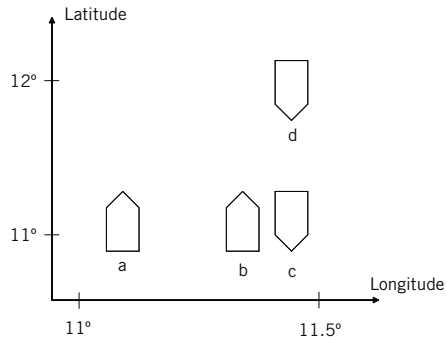


Figure 3.1: Vessel similarity example.

Consider the four vessels, depicted in Figure 3.1. If the Euclidian distance is calculated on the basis of the latitude and longitude attributes, vessels *b* and *c* are the closest ones. If the vessels are in an area where the distance to the equator is considerable, vessel *a* is closer than vessel *d* to the vessels *b* and *c*. If the course attribute is included in the distance calculation, vessels *a* and *b* will be close to each other and vessels *c* and *d* will be close to each other. This is because the difference in course (180 degrees) dominates the calculation.

The problem with the non-linear correlation between the latitude and longitude attributes can be solved by projecting the coordinates into a linear x/y coordinate system. Normalization and adding a weight to each attribute can potentially solve the problem with one attribute dominating the calculation. However, it might be hard to set the weights in a way that makes sense to a human analyst, e.g., how much of a change in course corresponds to one longitude degree or one knot in speed?

This problem has been closely analysed by Gärdenfors [65] within the field of *Conceptual Spaces*. A conceptual space is defined as "*a set of quality dimensions with a geometrical structure*"- p. 24 [65]. The relation among the quality dimensions in a conceptual space can be either *integral* or *separable*. For an object with integral dimensions, it is impossible to assign a value to one dimension and not the others. An example of such an object is a RGB colour that consists of three dimensions: Red, Green and Blue. It is not possible to define a RGB colour object by only specifying a value for the Red dimension. Dimensions that are not integral can be considered to be separable. Examples of separable dimensions are the weight and geometry of an object. Many common distance metrices work well on objects with integral dimensions but very poorly on objects with separable dimensions.

Table 3.3: Pearson correlation between attributes in the AIS dataset.

	Latitude	Longitude	Course	Speed
Latitude	1	0.15168	-0.11088	-0.10134
Longitude	0.15168	1	-0.11286	-0.21631
Course over ground	-0.11088	-0.11286	1	0.05349
Speed over ground	-0.10134	-0.21631	0.05349	1

3.3.9.3 Mahalanobis Distance

The Mahalanobis distance measure can be used when attributes have different ranges and variances. The requirement is that the distribution of each attribute is approximately Gaussian and the attributes are correlated [132]. To evaluate whether this is the case with maritime AIS data, two experiments were conducted. In the first, the Pearson correlation [113] between all combinations of the four attributes in Table 3.2 was calculated. A correlation value of 1 or -1 indicates a strong positive or strong negative correlation; a correlation value of 0 indicates no correlation. The result of the experiment can be found in Table 3.3 and shows a very weak correlation between the attributes.

The second experiment evaluates whether the data instances for each attribute follow a Gaussian distribution. This is done by simply plotting the histograms of each attribute. As Figure 3.2 and 3.3 indicate, the latitude and longitude attributes can be said to be approximately Gaussian distributed, while the course and speed attributes are not.

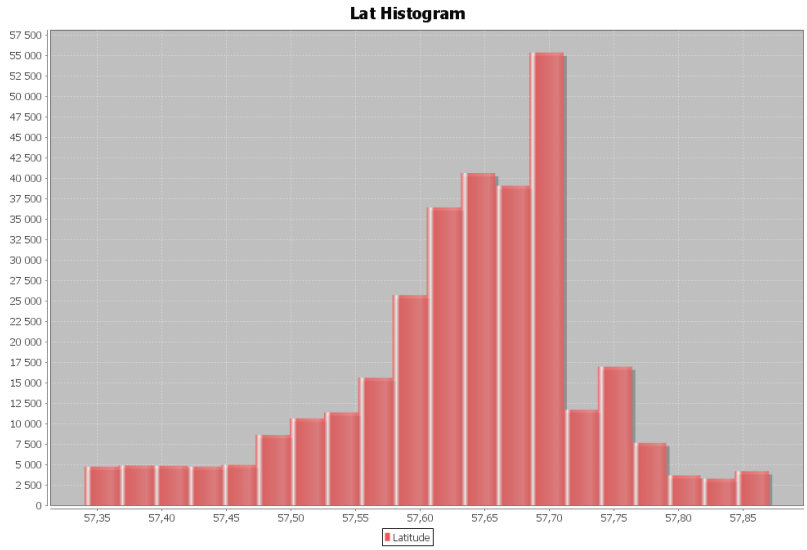
It is questionable whether the AIS dataset fulfills the correlation and distribution conditions of the Mahanalobis distance function. Based on these observations, it can be concluded that the Mahalanobis distance measure is not well suited for datasets like the AIS dataset.

3.3.9.4 Distance Measure: Conclusions

It is possible to calculate both Euclidian and Mahalanobis distances between AIS data instances. Calculating pairwise distances is computationally expensive, especially in the case of the Mahalanobis distance, where a covariance matrix also needs to be calculated. Moreover, the conditions for the Mahalanobis distance measure cannot be completely met, and it can be hard to find a weighing scheme that makes sense to a human analyst. Therefore, anomaly detection techniques based on the Euclidian or the Mahalanobis distance measures might not be suitable for the surveillance domain, if the datasets have the same properties as the AIS dataset used in these experiments.

3.4 Summary and Discussion

Based on the analysis of previous methods for anomaly detection in the surveillance domain, the SeeCoast approach is the most attractive. However, the description of the method lacks the degree of detail needed for an implementation. This might be due to proprietary reasons from the company responsible for the method. It should be possible to use the same type of representation in other approaches. The representation in SeeCoast does not rely on a distance measure and, therefore, does not have the same issues as the representation in the AADP method. The BALDUR, ARAD and HHMM methods use the probability for transitions between different states to detect anomalous transitions. This idea could be used together with the representation in SeeCoast. The GMM and KDE approaches have some strengths and can be useful as benchmarks when evaluating new methods. Based on the Anomaly Detection section in the background, and the fact that input data is unlabelled, the most suitable anomaly detection methods for the surveillance domain are the techniques based on unsupervised learning, such as clustering, statistical and nearest neighbour-based techniques. Both clustering and nearest neighbour-based techniques rely on a distance metric, hence, they are not the most appropriate techniques for the surveillance domain. Spectral techniques such as PCA could also be used, but the dimensionality of the surveillance data is already quite low, which makes it unnecessary to use dimensionality reduction techniques.



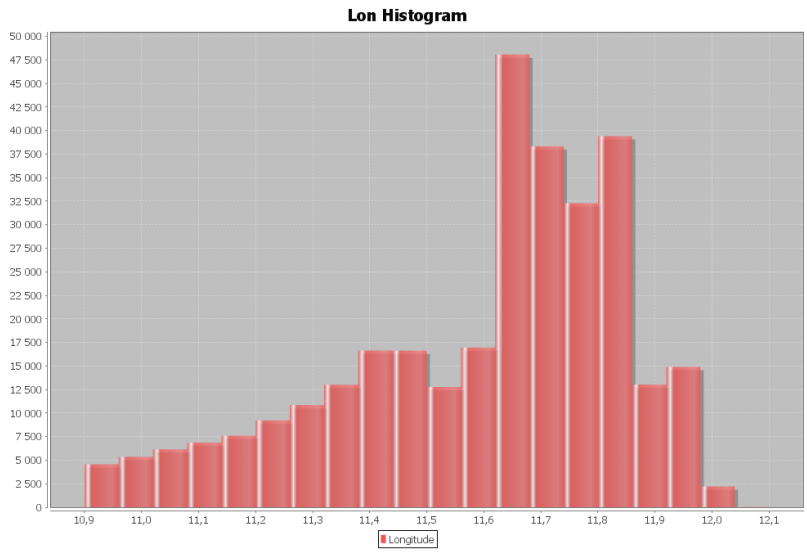


Figure 3.2: Histogram plots for the latitude and longitude attributes in the AIS dataset.

Figure 3.3: Histogram plots for the course and speed attributes in the AIS dataset.

Chapter 4
The State-Based Anomaly Detection Method

This chapter addresses research objectives two, three and four. The content of the chapter is largely based on the following publications[1]: Brax et al. [29, 28], Brax and Niklasson [26]. The main contributions of the chapter are:

- A new method for anomaly detection called *The State-Based Anomaly Detection* (SBAD) method
- A performance comparison between the SBAD method and a method based on Gaussian mixture models on outdoor video surveillance data.
- An evaluation of the SBAD method on maritime data from an airborne radar system.
- Results of an evaluation of data-driven anomaly detection used together with knowledge-based detection.

4.1 Finding Behavioural Anomalies in Public Areas Using Video Surveillance Data

The work in this section was part of a project called B&T (Business and Technology) Anomaly Detection, which was an internal research and development project at Saab AB. The overall goal of the project was to evaluate the feasibility of building an automatic video surveillance system based on anomaly detection. Outdoor public areas constituted the scene of interest and CCTV cameras provided the data.

[1]Section 4.1.5 is based on the co-author, Rikard Laxhammar's contribution to the [28] paper.

4.1.1 Introduction

Conventional systems for video surveillance usually rely on the manual interpretation of data presented on multiple screens [46]. Although such manual analysis by an experienced analyst can produce good results under ideal operating conditions (few sensors, small surveillance area, low data rate, low requirements for fast response etcetera), manual analysis can also be associated with problems, due to the inherent limitations of human analysts, these include: inconsistent analysis, missed detections due to distractions, boredom, lack of experience or fatigue, and low throughput during periods of high-volume activity. As a consequence, the usage of the large volume of data recorded by extensive networks of cameras operating in public areas such as public transportation facilities, town centres and parking lots, has mainly been restricted to forensic purposes. According to Dee and Velastin [46], the camera to screen ratio in the UK is between 1:4 and 1:30 and the ratio of operatives to screens is up to 1:16. This means that there are 4 to 30 cameras per screen in the surveillance centrals and as many as 16 screens per surveillance operator, i.e., at most, an operator has to monitor video information from 480 cameras. The goal of anomaly detection in the context of video surveillance in public areas is to aid human operators in detecting anomalies that correspond to illegal or dangerous human activities in real-time.

In this chapter, a new method for automating the process of analysing the behaviour of objects in video surveillance is proposed. The overall goal of the method is not to replace the operators, but to support them in their surveillance mission. This is accomplished by finding objects that deviate from what is considered normal, which must be done under run-time conditions and not afterwards, to enable the organization to initiate adequate responses to the observed behaviour. One part of such a system is the anomaly detector that uses models of normal behaviours to represent normalcy. These models are used to find anomalies in new observations. In order to build these normal models, a suitable representation of object behaviour, i.e., the change in object state over time is needed. There are many ways of representing object behaviour, ranging from just an ordered list of observations to high-level symbols. In addition to this, a recorded dataset is needed, in order to capture what is to be considered "normal" for the situation at hand. If a general structure for the representation can be identified, this structure should handle data for various situations and allow the generation of different models of normalcy, depending on the situation.

A good anomaly detection method must be able to learn normal behaviour and generalise to a "suitable" level, i.e., objects with similar behaviour should be represented in a similar way in the normal model and, if new objects arrive with similar behaviour, they should be classified in the same way. Consider two persons, A and B, who are walking along a pavement. Person A walks on the right side of the pavement and person B walks on the left side of the

pavement. The two persons can be considered to be behaving in the same way, even though they are not walking on the same path. A good normalcy modelling method should be able to generalise from examples, i.e., if the path of person A is modelled in the normalcy model and the model is used to classify the path of person B, the model should classify person B as normal, because it is similar behaviour to person A.

This chapter presents an approach for how some important parts of an automatic anomaly detection system can be designed and implemented. The main building blocks are a general structure for representing data, pre-processing of the data, normalcy modelling using situation dependent data and anomaly detection. The proposed representation and the normalcy modelling method are very intuitive compared to earlier approaches ([121, 85, 77]) and have a very low computational complexity, when generating the normal model and detecting anomalies. The representation also supports iterative updates of the normal model while, for example, the approach in Kraiman et al. [85] requires the normal model to be rebuilt from scratch, to add new observations to it. The SBAD normal model also has a high level of explainability for the user, which means that the user can see exactly which states have caused the anomaly.

In the experiment presented in this chapter, two approaches for anomaly detection are evaluated. The first approach is based on discrete states and second is based on continuous momentary track features. This approach has previously been used in the maritime domain [87]. The two approaches are evaluated using real-world data, recorded in a demonstrator system, i.e., a system constructed for evaluating automatic video surveillance. The reason for the construction of such a system is that no operative surveillance systems were available to use for these kinds of experiments.

4.1.2 The State-Based Anomaly Detection Method

Many of the previous approaches [85, 121, 87] for anomaly detection use models based on continuous values, such as Gaussian mixture models. Moreover, many of these only consider momentary observations and it is not obvious how to incorporate the temporal and contextual aspects of the data. Yet, in many anomaly detection applications, it is desirable to find objects that not only deviate in the basic attributes such as position and velocity, but also objects that deviate with respect to contextual information and high-level behaviours, such as relations to other objects over time and relations to the environment. This class of anomalies can be regarded as behaviour anomalies and, in this context behaviour is defined as the *change in state over time*.

When building models of normalcy, it is common to extract the basic attributes in the data, such as position and velocity, to describe the objects' momentary state. An alternative approach is to use a more abstract representation, which can be accomplished by pre-processing the basic attributes. For example, it is sometimes not that important whether an object moves at 12.2 m/s or 12.4

m/s. Instead, it might be more important to know whether the object is moving above or below 10 m/s. In this case, it could be suitable to create a new attribute called, for example, SpeedState, which can have the discrete values fast or slow, corresponding to a speed above or below 10 m/s. An advantage with this kind of representation is that the object's speed over time can be modelled as a sequence of SpeedStates instead of a sequence of absolute speed values. This way, the data needed to describe how the speed develops over time can be decreased. The behaviour of an object can be expressed as: "object X was in (SpeedState=low) for T_{n1} time steps and then it changed the speed to (SpeedState=fast) for T_{n2} time steps". Various time-series analysing methods could be used to further process the state sequence.

SpeedState can be regarded as an *atomic state class*. If the SpeedState is combined with another atomic state class, e.g., a discretised position state, the result is combinations of state classes that can be handled in the same way as atomic state classes. The combination of atomic state classes describes the behaviour of an object at a higher level than the atomic states class and is called *composite state class*es. Sequences of atomic and composite states can be analysed in the same fashion and the sequences, can for example be used to classify or group objects together, based on their behaviour.

The discrete state representation is the basis of the State-Based Anomaly Detection (SBAD) method and can be used for both contextual and kinematic information. More formally, the *atomic states* are represented by *discrete random variables* $Y_1, ..., Y_n$ all of which have a state space Ω_Y. Let y be any state in the state space Ω_Y, i.e. $y \in \Omega_Y$. Each $Y_i, i \in \{1, ..., n\}$, represents a *discretised attribute* from the kinematic or contextual information. As an example, Y_i can denote the course of the object of interest, where the state space is $\Omega_Y \triangleq \{north, south, east, west\}$. The state space Ω_{Y_i} for each Y_i is set by a domain expert, on the basis of what can be expected in the dataset from the given domain. This way *expert knowledge* can be included in the representation. The discretisation parameters can also be set by using automatic methods, such as equal width or equal frequency, described by Liu et al. [94]. The discrete state spaces allow for a higher level of representation of continuous attributes. Atomic states $Y_i, i \in \{1, ..., n\}$, can be combined into a *composite state* $\mathbf{Y} \triangleq Y_1 \times ... \times Y_n$ that captures the dependencies between atomic states. For example, stating that a vessel is travelling north is not very informative, but stating that a vessel is travelling north on a certain position, can be much more informative.

An example of a two-dimensional composite state with course and position in time step t might be $\mathbf{Y}_\mathrm{t} = [pos_1, north]$. This indicates an object whose position falls into a grid cell labelled pos_1 with a course falling into the atomic course state $north$.

By utilizing a *training dataset*, consisting of a number of tracks, a normal model that captures different aspects of the domain of interest can be built. Each track is represented as a sequence of observations of an object, and each

observation has a number of attributes, e.g., time-stamp, position, speed, course and so forth. In the SBAD pre-processing step, the attributes in each observation of a track are mapped to the corresponding atomic and composite states $\langle \mathbf{y}_1, ..., \mathbf{y}_t \rangle \in \Omega_{\mathbf{Y}_1 \times ... \times \mathbf{Y}_t}$, In order to capture the normality with respect to composite states, the *relative frequency* of the *composite state occurrences* in the training dataset can be utilized, i.e.:

$$p_{Occ}(\mathbf{y}) \triangleq \frac{n(\mathbf{y})}{\sum_{\mathbf{y} \in \Omega_{\mathbf{Y}}} n(\mathbf{y})}, \tag{4.1}$$

where $\mathbf{y} \in \Omega_{\mathbf{Y}}$ and $n(\mathbf{y})$ denotes the number of observations of $\mathbf{y}$.

The use of composite states is an intuitive approach of representing *joint attributes*. As stated earlier, kinematic attributes in the surveillance domain are often joint, e.g., the latitude and longitude positions must be used together to be of any use. The same is true for the speed and course attributes, which should be used together with the position. However, it is not always necessary that all attributes must be used. It might be perfectly all right to only use the position and the speed for some applications. Another advantage of using a discrete representation is that some of the uncertainty in the data can implicitly be handled when discretising attributes, e.g., if an object's speed fluctuates a little, it might not affect the corresponding atomic state. However, if the fluctuation is around the boundaries between two state classes the fluctuation can be a problem. Therefore, the number of state classes in each atomic state and their corresponding boundaries can have a great impact on the final anomaly detection result. Also, the precision of an attribute represented in a discrete way is much lower than the corresponding continuous representation. Hence, some information is lost when the input data is discretised. The advantage of the representation is that large datasets can be effectively represented with low memory requirements. The disadvantage is that the lost precision might, in some cases, be required for separating different object behaviours and some anomalies might therefore be impossible to detect. Another potential problem with this kind of representation, as well as with continuous representations, is that the state space could be sparse, when the number of attributes in the data increases. The discrete representation is based on ideas from the representation proposed by Rhodes et al. [109]. The main difference is that the representation used by Rhodes et al. [109] uses fixed attributes, i.e., only position, speed and course, while the representation proposed in the SBAD method is dynamic and can use an arbitrary number of attributes, which can be combined in different ways, i.e., multiple composite states with different subsets of the atomic states can be defined. The normal model built from composite state occurrence statistics, can be seen as a multi-dimensional histogram [100, 83].

4.1.3 Experimental Setup

In this section, two approaches for building normal models and detecting anomalies are evaluated. The first is based on the discrete state representation; this approach uses context information and relations between objects to create a richer normal model compared to normal models that only include kinematic information. The idea is that this approach should enable the system to find both simple and complex anomalies. Simple anomalies only involve one object, while complex anomalies involve more than one object. The second approach is based on continuous momentary track features learnt from data, similar to the approach used in [87]. This approach does not support the inclusion of context information and relations between objects, hence, it might have some problems detecting complex anomalies.

4.1.3.1 Dataset Description

The dataset used in this experiment was collected from the demonstrator system, which was installed on top of a building. The demonstrator system consists of two camera pairs[2] looking downwards. The left camera pair was directed towards a traffic junction and the right cameras pair mainly covered a small parking lot (see Figure 4.1). The azimuth sector covered by the cameras was roughly 90 degrees and the distance to the centre of the scene is approximately thirty meters. Data was collected during day-light only. The cameras were connected to a laptop PC placed inside the building. An off-line video tracker was used to extract tracks from the recordings. With some optimizations of the video tracker, it should be possible to run in real-time, but in this experiment, real-time tracking was not the main focus. The video tracker outputs time-stamp, track id, position (x, y), velocity (x, y) and object size for each tracked object.

Normal Behaviour of Objects in the Data

The scene was mainly populated with cars, bicycles and pedestrians. The movement of cars and bicycles was constrained to the road network. Although the motion of people involves more degrees of freedom, well established walking paths can be identified. The number of established tracks recorded during one day was approximately 1000. It was assumed that a few days of training data was enough to establish a normal model for the scene used in the demonstrator. In data-driven anomaly detection, the training data is assumed to be free of anomalies. This is obviously not true when dealing with real-world data. Analysis of false positives in the validation dataset reveals a number of anomalies in the training data. However, the number of false positives is quite small

[2]The use of cameras in pairs enables greater tracking accuracy by using the distance information extracted from the stereo images. The technique is similar to the one used by TRACAB to track football players [9].

Figure 4.1: Images from the left and right camera pair.

compared to the number of normal observations and should not impact the performance of the normal model too much. The fact that there are assumed to be only positive examples (with the label normal set to "true") in the training data, limits the choice of algorithm the class of algorithms that does not require labelled training data with both positive and negative examples.

Anomalies in the Data

To evaluate the performance of the anomaly detection approaches, a one-hour sequence of test data with a number of deliberately inserted anomalies was recorded. This dataset was manually labelled. Both simple single object anomalies and more complex ones involving multiple objects were recorded. Examples of single object anomalies recorded in the test data are illustrated in Figure 4.2 and include:

- A person walking on the grass where people normally do not walk (1 instance).
- A person walking on the grass instead of on the pavement (2 instances).
- A person running where people usually walk, one instance from left to right and one from right to left (2 instances).
- A person staying in the scene for a long period of time (1 instance).

The scene was also prepared with some examples of complex anomalous behaviour, involving more than one object. These anomalies might require the use of more advanced anomaly detectors, which need to handle more attributes and/or temporal aspects. Examples of complex anomalies include:

pick pocketing scenario involving four persons, where two persons bump into each other, a third takes the wallet from the victim and delivers it to a fourth person (2 instances).

snatching, where a person catches up with a victim, grabs his bag and runs away (2 instances).

pursuing, where a person waits for another person and starts to pursue him when he has passed by (2 instances).

Notice that the scene used in this experiment is not very crowded. The pick-pocketing scenario is more likely to occur in a much more crowded area where it would be much harder to detect. A more crowded area would put a lot more stress on the tracking and anomaly detection systems.

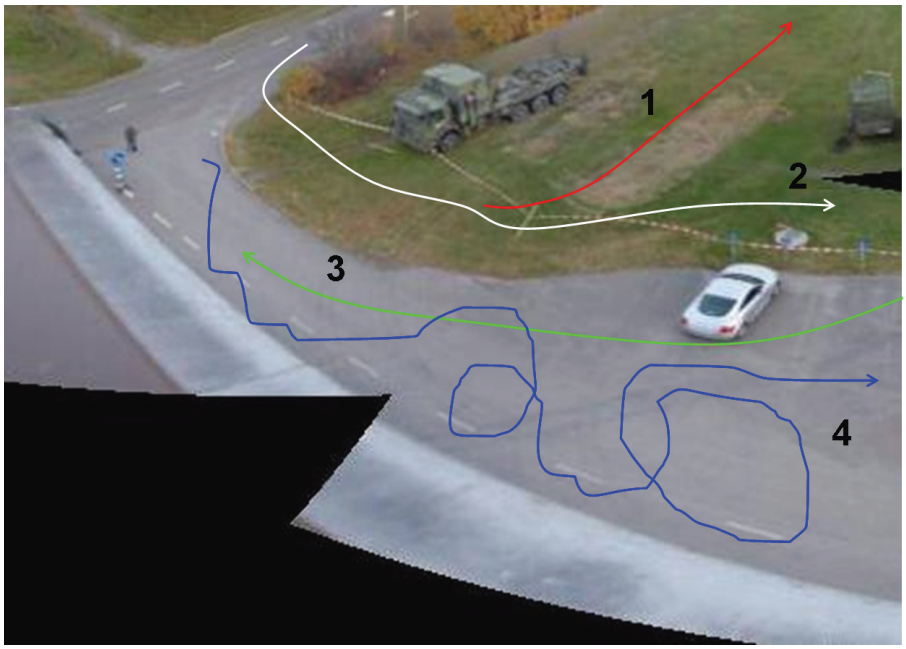


Figure 4.2: Illustration of single object anomalies.

Table 4.1: The number of tracks and observations in the four experimental setups.

	Training dataset		Validation dataset	
	Number of tracks	Number of observations	Number of tracks	Number of observations
1	3720	412592	1824	191977
2	4259	470909	1285	133660
3	4110	438514	1434	166055
4	4543	491692	1001	112877

4.1.3.2 Training, Validation and Test Dataset

The demonstrator system was used to record four days of unlabelled training data and one hour of manually labelled test data with anomalies. This data was used for the evaluation of the anomaly detection approaches. The test dataset includes 12 anomalies. The four days of training data were used to make 4-fold cross validation. This means that data from three days was used for training and data from the fourth day was used for validation. Four datasets were created with the different days as the validation day. The four sets are called Setup 1, Setup 2, Setup 3 and Setup 4 (see Table 4.1 for detailed statistics of the datasets).

4.1.3.3 Pre-processing

In anomaly detection as well as in other data mining applications, pre-processing is usually the most important step [104]. It is in this step that a suitable representation for the data is developed; the representation will later be used by various algorithms. The pre-processing is highly dependent on the class of algorithms used. For example, if the algorithm work on discrete data and the sensor information are continuous, the data must be discretised before it can be used.

In the pre-processing, objects that for some reason (usually by the trackers' dead-reckoning algorithms) end up outside the scene are removed. All cars in the data are also removed by using the *object_size* attribute from the tracker. The main reason for this is that the tracker used in this project was not optimized for tracking cars and, hence, generated a lot of ghost tracks around the actual cars.

4.1.4 Anomaly Detection Based on Discrete States

The first approach for anomaly detection is the SBAD method, which is based on a state-based representation of the track data. This kind of representation enables the possibility to include additional information, besides kinematic

data, in the normal model. The hypothesis is that if context and other a priori knowledge are to be included into the normal model, the model can be made simpler without sacrificing classification performance. In this experiment, GIS information about the scene is available; this information includes which areas are assigned to roads, grass and pavements. If this information can be include in the normal model, the anomaly detector would be able to distinguish between objects moving on different grounds. Furthermore, information about relations between objects can also be included in the normal model, to be able to detect anomalies involving more than one object, e.g., people following each other, such as in the case of snatching.

4.1.4.1 State Classes

Instead of using the attributes from the tracker directly, a more abstract or high-level description of the data is used to be able to incorporate a priori information about the domain. In this experiment, five atomic state classes are used; *SpeedState*, *CourseState*, *RelationState*, *EnvironmentState* and *PositionState*. This is mainly due to the availability of information. The SpeedState, CourseState and PositionState are based on information from the video tracker. The RelationState was used to be able to capture complex anomalies where the relations between individual objects affect their behaviour, c.f. the snatcher scenario. Due to the availability of a map over the area, the idea behind the EnvironmentState was to be able to build a less complex model, if the relation to the environment could be modelled directly instead of having to be learned from data.

The *SpeedState* is built from the speed of each observation. There are four different classes; *Stopped*, *Slow*, *Medium* and *High*. The boundary between the classes is chosen to reflect stationary objects, people walking, people running and vehicles respectively. The velocity from the tracker is used to set the SpeedState. The choice of classes is context dependent and should be set with regard to domain expert knowledge.

The *CourseState* represents the absolute course of an observation and is divided into eight classes; *north*, *northeast*, *east*, *southeast*, *south*, *southwest*, *west* and *northwest*. Each class is 45 degrees wide. The *north* class is between 337.5 and 22.5 degrees and so on. The course is calculated using the x and y velocities from the tracker.

The *RelationState* defines the relation between an object and the closest other object. If the closest object is within a user defined distance, the RelationState is set to one of the following classes; *inFront*, *behind*, *left* or *right*. If the closest object is outside the threshold, the RelationState is set to *undefined*. For example, if an object moves east and another object is west of the first object the relation from the first object to the second one becomes *behind*, and so forth. The *RelationState* is calculated using the position and course of the objects in the relation.

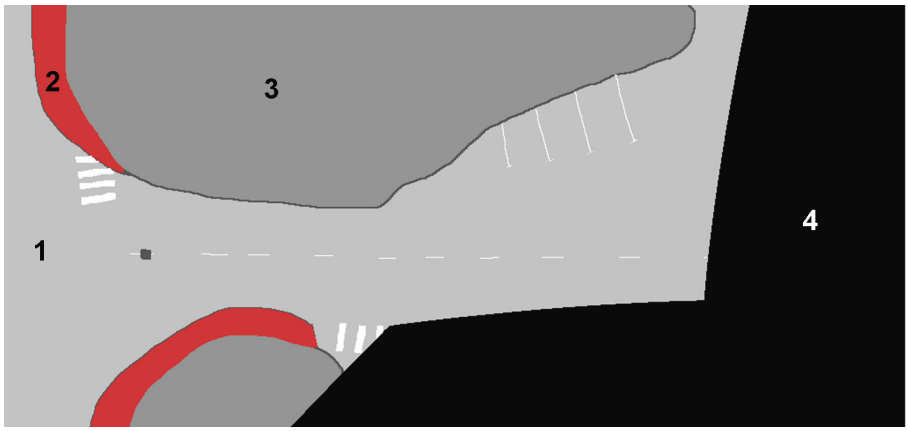


Figure 4.3: Map used for calculating environment states.

The *EnvironmentState* is used to incorporate geographical information about the scene. This state describes the relation between an object and the underlying environment, which assumes a priori information about the surveyed area. In our experiments we used the map depicted in Figure 4.3 to classify the *EnvironmentState* as: onRoad (1), onPavement (2), onGrass (3) or undefined (4).

The *EnvironmentState* basically describes on which ground an object is moving, and is calculated by using the x and y position of the object in conjunction with the map.

The fifth state class is the *PositionState*, which is just a discrete representation of an object's x and y position. After some initial tests, the parameters of the grid were set to 4 rows and 10 columns. This results in a total of 40 different position classes. In the scene used in the experiments, the height of one cell, in meters, corresponds to the width of the road at the narrow parts. The height and width of the grid cells, in meters, are constant.

These five state classes are called atomic states; in this experiment they are combined to a composite state. It is also possible to make other composite states that have combinations of two to four atomic states. The *CompositeState*, can be describes as Equation 4.2:

$$\mathbf{Y} \triangleq Y_1 \times ... \times Y_n, \tag{4.2}$$

where $\mathbf{Y}$ denotes the composite state and Y_i denotes an atomic state.

The datasets used for training, evaluation and for testing are all preprocessed into atomic and composite states before they can be used for building the normal model or detecting anomalies.

4.1.4.2 Building the Normal Model

When building the normal model, the training datasets are used to generate atomic and composite states for each observation. The relative frequency of each unique composite state is used as the normal model, i.e., the number of observations in the training data that match a specific composite state is a metric of how normal that composite state is. At least one observation in the training dataset is needed to generate a composite state in the normal model. The computational complexity of building the model from pre-processed composite states is $O(n)$, where n is the number of observations in the dataset. The pre-processing also has $O(n)$ complexity except for the step generating the RelationState. This step has at most $O(n^2)$ complexity, if a brute force search for finding the closest object to a specific object is used. This complexity could be reduced by using a spatial indexing method such as Quad-Trees [120]. The way the normal model is built allows for incremental updates of the model without rebuilding it from scratch.

4.1.4.3 Detection of Anomalies

The anomaly detection is quite simple; the system just generates atomic and composite states for the new observation from the test dataset and then tries to find the corresponding composite state in the normal model. If the frequency of that state is below a threshold, the observation can be classified as anomalous; otherwise it is classified as a normal observation. In practice this is done by a simple table-lookup and should therefore be possible to do in real-time for a large number of objects.

4.1.5 Anomaly Detection Based on Continuous Momentary Track Features

The second approach proposed and evaluated in this experiment is based on the statistical modelling of continuous momentary track features. The proposed model has been adapted from previous work within sea surveillance [87] where the momentary position and velocity vector in four-dimensional Cartesian coordinates of individual vessels were statistically modelled. However, the feature model was extended to also include the accumulated time by which the object was tracked, i.e., the momentary track age at each particular time instance. Thus, each data vector has five features corresponding to the: *x-coordinate*, *y-coordinate*, *x-velocity component*, *y-velocity component* and *momentary track age*.

Generally speaking, we could either have these five features span a complex continuous five-dimensional feature space, or we could have a number of separate low dimensional feature spaces, e.g., one for each type of feature; two two-dimensional models for the position and velocity, respectively, and one

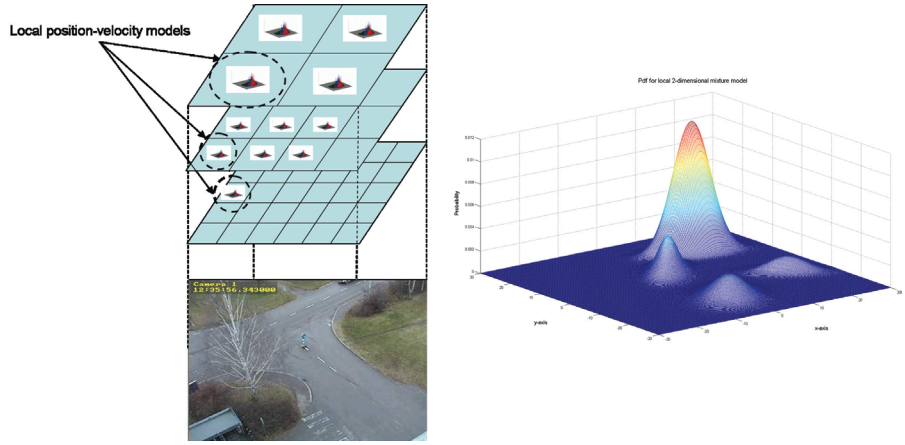


Figure 4.4: Hierarchical grid with multiple spatial scales and an example of the PDF of a 2-dimensional GMM. Courtesy of Rikard Laxhammar.

one-dimensional model for momentary track age. A complex feature model is theoretically attractive, because it captures correlations between the different feature types, such as position and velocity. However, it may imply practical constraints regarding the computational feasibility. Moreover, how to normalize the different features during normalcy learning and anomaly detection it may not be obvious. In this application, it is assumed that there are potentially interesting anomalous correlations between the position and velocity of track data. However, correlations between the position-velocity vector and the momentary track age were assumed to be of less interest. Therefore, in order to suppress computational complexity, two non-correlated models were suggested; a four-dimensional model representing the momentary position and velocity vector in Cartesian coordinates, and a one-dimensional model representing the momentary track age.

In order to accurately model local features of data and, at the same time, suppress complexity, the surveillance area was discretised into a uniformly sized grid where each cell models local features of data within the confined area [87]. The grid size is determined, more or less arbitrarily, on the basis of the application and the surveillance area. However, in order to better capture local and semi-local features related to position and velocity, we extended the previous approach by introducing a hierarchical grid at multiple spatial scales (see Figure 4.4). For each cell at each scale, a local model for normal position-velocity vector values was used. Momentary track age is not assumed to be spatially dependent to the same extent as velocity and was therefore captured by a single global model valid for the entire area.

4.1.5.1 Statistical Modelling and Learning

Analogously to Laxhammar [87], normal values of the position-velocity vector and the track age scalar are statistically modelled by multivariate- and univariate Gaussian Mixture Models (GMM), respectively. The probability density functions (PDF) of the mixture models are used for calculating the degree of normalcy of a data vector. Assuming that the data vector is normal (i.e., it constitutes a sample from the distribution of normal data vector values), its likelihood can be calculated from the PDFs.

In order to obtain the PDFs, the parameters of the corresponding GMMs need to be estimated, i.e., the number of Gaussian mixture components and the mean value, as well as the covariance matrix and weight value for each of them. This is carried out using a greedy version of the Expectation-Maximization (EM) algorithm that incrementally finds the optimal model parameters that maximize the likelihood of a training dataset reflecting normalcy [87]. The algorithm starts by estimating the parameters of the optimal one-component model using regular EM. It then inserts a new component and estimates the optimal parameters of the new two-component model using EM. These steps are repeated until the likelihood of a validation set decreases, i.e., when the algorithm starts overfitting the model to the training data by adding too many components.

4.1.5.2 Detection of Anomalies

When presented with a new data vector, the algorithm for anomaly detection first determines, for each spatial scale, which cell the observation belongs to; this gives us the corresponding local position-velocity GMM for each scale. Next, the local position-velocity likelihoods for each scale are fused by taking the maximum value. The fused position-velocity likelihood is then fused with the global momentary track age likelihood for the data vector, resulting in a single scalar value. At this point, the momentary likelihood could be compared with some anomaly threshold, immediately classifying it as either normal or anomalous. However, in the context of surveillance and tracking of humans, tracks that behave more or less anomalously over time are probably more interesting than momentary anomalous measurements which can be a result of noise in tracking data. Therefore, it is proposed that the momentary likelihood for a sequence of data vectors should be averaged within a sliding time window frame for each active track. If the average likelihood is below a certain user set threshold, an alarm is issued and the track is regarded as anomalously from the start of the window and further on, e.g., until an operator manually dismisses the alarm. If, for some observed data vector, no local normalcy model is available at for all scales (e.g., due to lack of training data in the area or the fact that the data vector lies outside the boundaries of the hierarchical grid) the momentary likelihood is set to zero.

Depending on the specific application, it may be appropriate to have multiple sliding windows at different time scales working in parallel, where each window has a fixed size (time interval) and threshold. For example, consider the case that we have a single, relatively small window with a relatively low likelihood threshold. Such a window would be effective for the early detection of rather distinct anomalies, but it might miss more subtle anomalies that develop over a longer time period. However, complemented with a relatively large window with a higher threshold, we might capture more interesting anomalies and at the same time reduce the level of false alarms. Nevertheless, during the experiments, a single window was considered to be adequate.

4.1.6 Experimental Results

In this section, the results of our experiments are presented together with information about how the performance was measured and the threshold settings used in the experiments.

4.1.6.1 Performance Measures and Parameter Estimation

Parameters and Threshold Tuning for the SBAD Method

Section 4.1.4.3 describes how to use the SBAD method to classify a single observation in a track as either normal or anomalous. To classify the complete track or, more precisely, the object the measurements of which resulted in the track being anomalous, we obviously have to use more than a single observation. Noise and uncertainties in the system can result in the observations of normal objects being classified as anomalous, which should not raise the alarm to the operator. In the SBAD method, the accumulated number of anomalous observations during the track's life-time is used to assess the degree of anomaly for a track. If the degree of anomaly reaches a user-defined threshold, the system will classify the track as anomalous. This threshold is context and application dependent and the user should be able to raise and lower the threshold depending on the current situation.

After an analysis of the distribution of the number of anomalous observations per track, it was concluded that the distribution approximately followed the normal distribution. Therefore, the mean and standard deviation can be used to set thresholds for when to classify a track as anomalous. The mean and standard deviation for all four validation datasets can be easily calculated; Table 4.2 illustrates the result.

Table 4.2: The mean and standard deviations of the number of anomalous observations classified as anomalous calculated for the four validation datasets and for all four sets combined.

Validation dataset	Tracks with anomalies	Anomalous Observations	Average number of anomalous observations per track	Standard deviation
Setup 1	14.36%	1.87%	13.67	26.01
Setup 2	8.64%	0.54%	6.56	8.98
Setup 3	16.53%	4.13%	28.95	70.62
Setup 4	14.19%	1.87%	14.88	29.86
Validation dataset 1-4 combined			17.66	45.24

As Table 4.2 illustrates, the mean and standard deviation for the validation datasets 1 to 4 are 17.66 and 45.24. To obtain a reasonable number for the threshold regarding how many anomalous observations a track must have before we classify it as anomalous, the values from one and two standard deviations from the mean in the validation dataset are used. In this case, the thresholds used for detecting anomalies will be 63 (62.91) and 108 (108.15) respectively. Note that these two thresholds correspond to the object being anomalous for 2.5 and 4.3 seconds (the video tracking system outputs 25 observations per second).

Two different experiments are conducted with the SBAD method. The first one uses a normal model with composite states built from all five atomic states presented in Section 4.1.4.1. The second experiment only includes atomic states related to position, speed and course. This way, the use of contextual information, regarding detection performance, can be evaluated.

Parameters and Threshold Tuning for the Continuous Momentary Feature Model

In the experiments with the continuous momentary track feature model, the local position-velocity vectors were modelled at two scales where the number of grid cells was 8x4 and 10x5, respectively. As described in Section 4.1.5.2, the anomaly detection is based on averaging the fused data vector likelihoods of the sequence of momentary data vectors within a sliding window for each track, where the size of the window was set to 2s. If the logarithm of the average likelihood is below the anomaly threshold, the track is henceforth classified as anomalous, starting from the time point of the first data vector within the current window. Remember that, if the data vector belongs to a cell that lacks

Table 4.3: Anomaly detection results of the SBAD method using three different detection thresholds. The normal model includes contextual information. The results are averages of the four experimental setups.

Threshold	Simple anomalies found (average)	Complex anomalies found (average)	Total anomalies found (simple and complex)	True positives rate in total anomaly set	False positive rate (average)
37	7/7	5/5	12/12	100.00%	1.35%
63	6/7	5/5	11/12	91.67%	0.83%
108	5/7	3.5/5	8.5/12	70.83%	0.49%

a local position-velocity model or, if it belongs to no cell at all (i.e. outside of grid boundaries), the corresponding likelihood is set to zero.

The tuning of the anomaly threshold is done individually for each setup 1-4 and guided by a tuning parameter that controls the rate of anomalous track detections in a specified tuning set. For these experiments, the set of simulated simple anomalies was chosen as the tuning set and the tuning parameter was varied. Therefore, setting the tuning parameter to 1 would result in an anomaly threshold that detects all the simulated simple anomalies.

4.1.6.2 False Positives and Detected Anomalies

Performance Evaluation of the SBAD Method

To assess the performance of the state-based normal model, the four validation datasets and the tracks from our 12 anomalies are used; the results of the evaluation with a normal model including contextual information are shown in Table 4.3. The use of the four normal models, revealed that the anomalous track with the least number of anomalous observations had 37 anomalous observations, which is the threshold we have to set to detect all anomalies in our anomaly dataset. We decided to include this number as a threshold in the evaluation to see the number of false positives when we set the threshold to detect all anomalies. As expected, the number of false positives in the validation datasets decreases when the detection threshold is raised. Table 4.4 shows the result of the experiments without including contextual information in the normal model. The results demonstrate that, if contextual information is included, more of the complex anomalies are found. The cost of this is a higher false alarm rate and fewer simple anomalies found.

Table 4.4: Anomaly detection results of the SBAD method using three different detection thresholds. The normal model only includes kinematic information. The results are averages of the four experimental setups.

Threshold	Simple anomalies found (average)	Complex anomalies found (average)	Total anomalies found (simple and complex)	True positives rate in total anomaly set	False positive rate (average)
37	7/7	3.5/5	10.25/12	85.42%	0.54%
63	6.25/7	2.25/5	8.5/12	70.83%	0.40%
108	5.25/7	1.25/5	6.5/12	54.17%	0.25%

Table 4.5: Results from the continuous momentary track feature model.

Anomalous tracks detected from the set of simple anomalies	Rate of true positives in anomaly set (tuning parameter)	Average rate of false positives in validation sets
7/7	100.00%	0.16%
6/7	85.71%	0.16%
5/7	71.43%	0.11%
4/7	57.14%	0%

Performance Evaluation of the Continuous Momentary Track Feature Model

The performance evaluation of the continuous feature model is done in a similar fashion as for the state-based model. In essence, the rate of false positives in normal data (corresponding to false/unwanted alarms during a real-world application) is estimated as a function of the (known) rate of true positives (corresponding to true/wanted alarms) in the set of simulated simple anomalies plus one of the snatching scenarios (the other complex anomalies were not tested). Technically, this is done by calculating the average rate of false positives in the validation sets for different anomaly thresholds, where the anomaly threshold for each validation set is determined by the threshold tuning described earlier. Thus, the rate of true positives is equivalent to our anomaly threshold tuning parameter in these experiments. See Table 4.5 for the results of the evaluation of the continuous momentary feature model with three different thresholds.

It is hard to make an exact comparison of the two approaches, since the pre-processing and threshold settings differ due to the nature of the methods used to build the normal models and detect the anomalies. Nevertheless, some

metrics can be used to compare the performance of the two approaches. The most important metric is how many of the anomalies, present in the test dataset, were each of the approaches able to find. Another metric is the number of false positives in the validation datasets. As the results indicate, the GMM approach is able to find the same simple anomalies as the SBAD method, but with a lower false alarm rate.

4.1.7 Discussion

The results of these initial anomaly detection experiments are promising, without too much fine tuning and optimization of parameters, both approaches manage to find most of the anomalies without a high false positive rate. The false alarms rate was lower than 1% most of the time, which translates into about two false alarms per hour.

The state-based approach found all five complex anomalies, when using contextual information in the normal model. Without contextual information it was harder to find the complex anomalies. Most of the simple anomalies were found by the state-based approach. The momentary track feature approach found the snatcher anomaly and had problems with one of the simple anomalies that the state based approach also had problems finding.

Two troublesome anomalies were the ones with the person running where people usually walk (3) from left to right and one instance of the anomaly with a person walking on the grass and not on the pavement (2). After analysing these two anomalies, we found that they are very similar to the normal behaviour of people riding their bikes.

Two different ways of classifying a whole track as anomalous were used. In the first approach, we used accumulated anomalous observations during the whole track life. This way of classifying tracks is dependent on the length of the track. If the track is very short, there might not be enough observations to be classified as anomalous at all and, if the track is very long, it might be wrongly classified just because of the fact that it has been in the scene for a long time and recorded many anomalous observations. The second approach uses a sliding window; this way the classification is not dependent on the length of the track. Tracks classified as anomalous can be re-classified as normal if the object starts to behave normally again. This property might or might not be desirable, depending on the application domain. The two approaches for classifying tracks could be used with either one of the anomaly detection methods, e.g., using the SBAD method together with a sliding window. This should be evaluated in future experiments.

Although the two approaches presented in this section had a similar level of performance, they both have their pros and cons. One advantage of the SBAD approach is the low computational complexity, both when building the normal model and when detecting anomalies. Another advantage is that the model is transparent to the user, i.e., the user can see exactly in which state-classes an

object deviated and it can detect all the complex anomalies. It is also possible to incrementally update the normal model with new observations, without rebuilding it from scratch. A disadvantage is that the SBAD method requires many parameters to set, and it is not obvious how to find the "optimal" parameter settings. In addition, the SBAD approach had a higher level of false positives. The GMM approach requires less a priori knowledge and has fewer parameters to be set. Furthermore, the computational complexity when building the normal model is much higher than the SBAD approach, but the complexity when detecting anomalies is low. However, the model must be rebuilt from scratch, to add new data to the normal model.

By comparing the two approaches, we can conclude that, if we have some a priori knowledge about the scene we can build a much less complex model and still have similar overall performance. If no a priori knowledge is available, the same level of performance can be achieved by increasing the computationally effort, at least when dealing with simple anomalies.

Another important issue is the threshold settings. How much must an object deviate from the normal model before it can be classified as an anomaly? This is of course context dependent. If the operator is in a stressful situation, the thresholds might be raised to prevent the operator from being overloaded with any more information, and, if the situation is very quiet, the operator can handle some false alarms, in which case the threshold can be lowered to increase the sensitivity of the system. Other aspects, such as response times for handling detected threats, might also be used when setting thresholds. If we are unable to respond to the threat, it might not be as important as other threats that we can respond to. The test dataset included quite a few anomalies, and it is therefore hard to draw any general conclusions from these experiments.

4.1.8 Summary

In this section, two different approaches for automatically finding behavioural anomalies in video surveillance data were presented and evaluated. Both approaches used recorded data, considered to be mostly normal, to learn statistical models representing normal behaviours of the objects in the surveillance data. There is a difference in the way the two approaches use a priori knowledge about the scene at hand. The first one incorporates as much a priori knowledge as possible, to be able to build a computationally simpler normal model, while the second approach requires a minimum of a priori knowledge, but needs more computations to build the normal model. Both approaches were evaluated using real-world data, recorded in a demonstrator system. The dataset consisted of four days of recorded, unlabelled data considered to be normal, and two hours of manually labelled data with both normal and anomalous observations. Both approaches were able to find most of the simple anomalies, without generating too many false alarms, and the SBAD method also found most of the complex anomalies.

In the future, it might be interesting to extend the SBAD model to not only use the probability of an observation being normal, but also the transition probabilities between states, which could minimise the number of false alarms. It might also be possible to combine the approaches and create Gaussian distributions of the state classes used in the state-based approach. In addition, the approach for detecting anomalies that develop over time should also be extended for the SBAD method.

The scene used for data collection in the experiment was sparsely populated, which might suffice for some applications, such as perimeter and area protection, but it is in crowded areas with many cameras, such as train stations and airports, that this kind of system can really help the operator. Highly populated scenes put a lot of stress on the video tracking system as well as on the anomaly detection. The approaches evaluated in this experiment should also be tested in crowded areas to assess their usefulness in more complex scenes.

4.2 Enhanced Situational Awareness in the Maritime Domain

This section describes an agent-based approach for situation management based on both knowledge-driven signature detection as well as data-driven anomaly detection. The approach is evaluated on a simulated dataset from the maritime domain. The main sensor used is airborne radar and the aim of this experiment is to evaluate the SBAD method on another application domain.

4.2.1 Introduction

Maritime Domain Awareness (MDA) is important for both civilian and military applications. An important part of MDA is the detection of unusual vessel activities such as piracy, smuggling, poaching, collisions and so forth. This is today usually done by highly skilled operators that constantly monitor and analyse the activity in an area of interest. There are, however, a number of problems associated with this approach. Today's interconnected sensor-systems provide huge amounts of information over large geographical areas which can cause the operators to reach the boundaries of their cognitive capacity and start to miss important events. In addition, boredom and fatigue are two other problems associated with the prolonged manual analysis of sensor information. The operators' personal experience and motivation may also affect their ability to find important events and their overall situational awareness [53].

This experiment is aimed at investigating how an automatic decision support and situation analysis can help the operator of an Airborne Early Warning and Control System (AEW&C) [127] (p. 548) to make the correct decisions at the correct time, i.e., to obtain an enhanced situational awareness. The AEW&C can detect aircraft and surface vessels at great distances, an ability

that makes it possible to survey large areas containing hundreds of vessels. The disadvantage is that it becomes very hard for the operator to find the interesting vessels that often are "hidden" in the noise originating from all the uninteresting ones. The decision support system is aimed at helping the operator to focus on important objects and thereby avoid information overflow [133]. An early warning of something, possibly suspicious, can enable the operator to be proactive and prevent unwanted situations arising.

4.2.2 Situation Management Approach

The situation management system is built on a multi-agent system (MAS) framework. It automatically analyses sensor information, to detect unusual activities and anomalies. The proposed system combines knowledge-based and data-driven anomaly detection. Knowledge-based anomaly detection is used to find known situations defined in advance by domain experts, while data-driven anomaly detection uses recorded data to build statistical models of normal vessel behaviour. The models are then used to classify new vessel observations as normal or anomalous. Moreover, the combination of the two approaches allows the system to detect previously known activities as well as unknown ones.

The two detection capabilities are implemented in a situation management system using a MAS framework called Java Agent Development Framework (JADE) [7]. Figure 4.5 shows an overview of the situation management system. The main components are the JADE Agents, where all processing and reasoning about vessel behaviour is carried out, the situation database that holds the current situation information, and the external communication package that handles all interaction with external systems.

There are two types of agents used in the system; administration and behaviour analysing agents. The administration agents are responsible for preparing the data and compiling the result obtained from the behaviour analysing agents. The *Director Agent* coordinates the work and triggers the analysing agents into analysing the current situation data. The *Compiler Agent* compiles the results obtained from the analyser agents and sends alerts to an operator via the external communication interface. The *Friendly Agent* searches for vessels that are considered to be friendly, and sets their corresponding affiliation to *friend*. This makes them easier to see for the operator of the Command and Control system. There are also four analyser agents; three of them are used for detecting specific, pre-defined scenarios, like *raid*, *smuggling* and *pursuing*, while the last one, the *Anomaly Agent*, uses data-driven anomaly detection to find new, previously unseen behaviours. The agents that are looking for pre-defined scenarios use a rule-based reasoning approach.

The main motivation behind the four analyser agents is to help operators of the AEW&C system to obtain an early indication of illegal behaviour such as piracy. A recent example of such activities is the Saudi oil tanker Sirius Star that was hijacked by pirates outside Kenya in November 2008 [8].

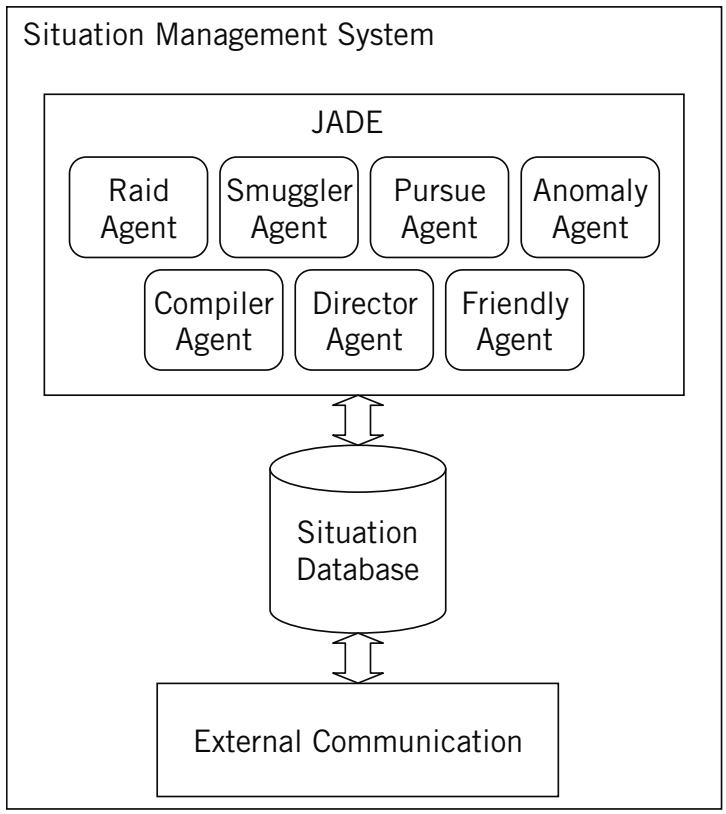


Figure 4.5: Overview of the Situation Management System and its components.

In this experiment, all three aspects of Situation Management (SM) are used. The investigative SM is performed by the anomaly detection where past situations are used to learn normal behaviours or entities. The control SM is related to the deliberative control loop where the operator is a part of the sensing-perception-problem-solving-affecting loop and uses the output from the Situation Model to make correct decisions. The predictive SM is performed for two reasons. The first reason is to find future anomalies; this is done by presenting future situations to the agents. The second reason is that some entities seldom get their attributes updated by sensory information. To get a complete situation where all entities have their attributes updated from the same time span, future attribute values must be predicted.

4.2.2.1 Pursue Agent

The pursue agent's main task is to analyse the behaviour of vessels and find those that pursue protected vessels. A vessel is labelled protected, if it has a large radar cross section, an AIS-transponder or it is transporting cargo deemed interesting to the operator. In the experiments described in this section, only the radar cross section condition is used for classifying protected vessels. This is due to the lack of AIS and cargo information in the dataset used for this experiment. The Pursue Agent has also been used in other experiments where the AIS and cargo information were available. Figure 4.6 shows the main components of the pursue agent. The agent searches inside the suspect area for vessels with roughly the same heading and speed as the protected vessel. When a matching vessel is found, it is put under special surveillance and, if it continues with the same behaviour for a period of time, it is tagged as a suspect pursuer and an alert is sent to the operator.

4.2.2.2 Raid Agent

The Raid Agent analyses vessels in the vicinity of a protected vessel and tries to find unknown craft approaching at high speed. If the approaching craft are inside the suspect area, they are tagged as suspected raiders and an alert is sent to the operator. If the craft leaves the suspect area, the suspect raider tag is removed, but should it enter the hostile area, then it is tagged as a hostile raider. In this case, the protected vessel is also be tagged as a suspect raider; this is mainly to indicate which vessel might have been raided if the hostile craft leaves sensor coverage. When a vessel is given the hostile tag, the situation management system does not remove the tag. This must be done by an operator after a manual analysis of the situation. Figure 4.7 shows the components of the Raid Agent.

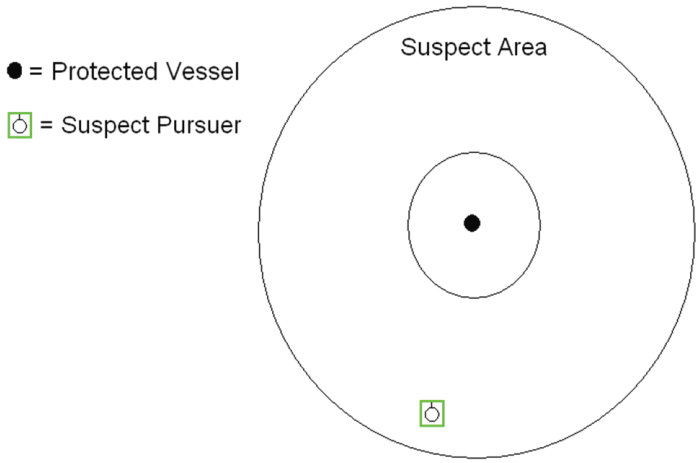


Figure 4.6: The main components of the Pursue Agent.

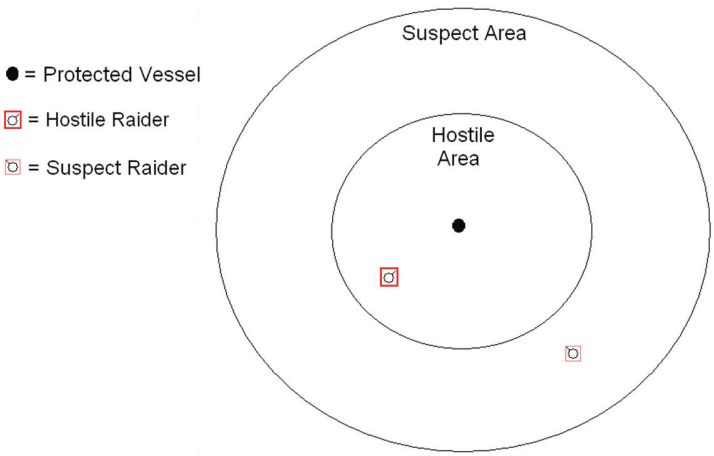


Figure 4.7: The main components of the Raid Agent.

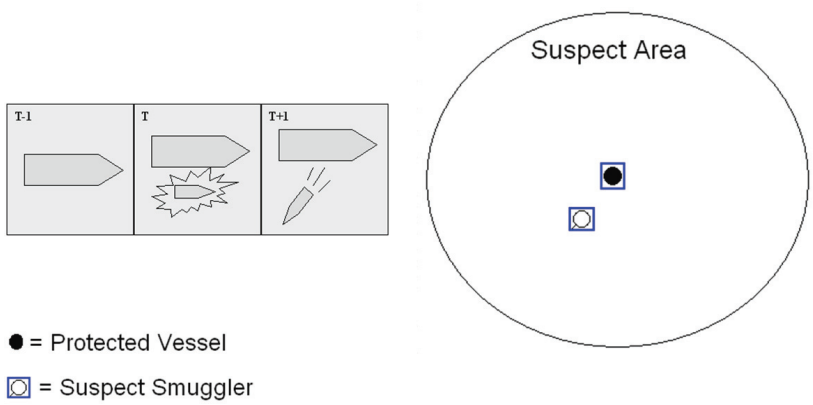


Figure 4.8: The main components of the Smuggler Agent.

4.2.2.3 Smuggler Agent

The Smuggler Agent is used to find a specific smuggling scenario. Figure 4.8 shows the main components of a smuggler scenario in which a large ship launches a small boat carrying contraband and heading for shore at high speed. The smuggler agent monitors an area around protected vessels called a suspect area. If a new vessel appears inside the suspect area, both the new vessel and the protected one are tagged suspect smuggler.

4.2.2.4 Anomaly Agent

The Anomaly Agent uses data-driven anomaly detection to learn the normal behaviour of vessels. The SBAD method is used to perform the data-driven anomaly detection and was previously used to find anomalies in video surveillance data (see Section 4.1). In this experiment, the method has been adapted to the maritime domain. In SBAD, the input data is discretised into a number of states, which provides a more high-level description of the state of a vessel. For this experiment, four different state classes are used to represent the state of a vessel; CourseState, RelationState, SpeedState and PositionState. CourseState is a discretisation of the course into eight different states. See Figure 4.9 for more information about the state. The RelationState describes the relation between two vessels close to each other. Possible state classes are inFront, behind, left, right and undefined. If the distance between a vessel (A) and the closest vessel (B) is above a threshold T_{Dist}, the RelationState is set to undefined. Otherwise, the relative position of vessel (B) from vessel (A) is calculated. For example, if vessel (A) moves west and another vessel (B) is east of the first vessel, the relation from the first vessel to the second one becomes behind. The SpeedState

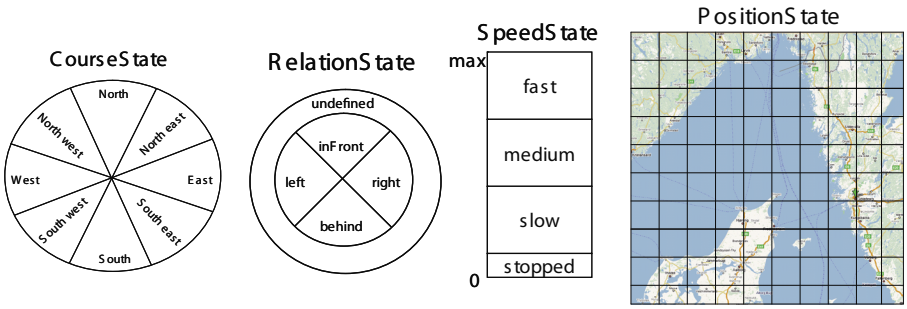


Figure 4.9: Visual representation of the four atomic state classes used by the Anomaly Agent.

is a descretisation of the vessel's speed. In this experiment, four different state classes are used; stopped, slow, medium and fast. The thresholds between the states are set by a domain expert to reflect the expected behaviour of the vessels in the area of interest. The last state is the PositionState discretisation of the latitude and longitude of the vessel into an n-by-m grid.

The four state classes above are called atomic states, and each one represents a basic attribute of a vessel. Moreover, the atomic states can be combined into a composite state which is a combination of atomic states.

One of the main concepts of data-driven anomaly detection is the normal model. It is a model representing what is normal in the system's domain. The normal model is built from pre-recorded data assumed to be normal. In SBAD, the normal model is built from the relative frequency of the enumerated composite states; see Section 4.1.2 for more information about how the normal model is constructed. When the normal model is built, it can be used for classifying new vessel observations as normal or anomalous. This is done by first generating the atomic and composite states for the observation, after which the normal model is used to obtain the relative frequency for the composite state from the observation. If the relative frequency is below a threshold T_{al}, the observation is classified as anomalous, otherwise it is considered normal.

4.2.2.5 Agent Priority

Since the analyser agents are independent of each other, multiple agents can make a classification of a vessel at a specific point in time. If this happens, the following priority is used in the Compiler Agent: Raid, Smuggler, Pursue, Anomaly and Friendly. The first three classifications are set by agents that have specific knowledge about the behaviour of the classified objects, while the two last classifications are more general and the agents making the classifications do not have a deep understanding of the vessels' behaviours.

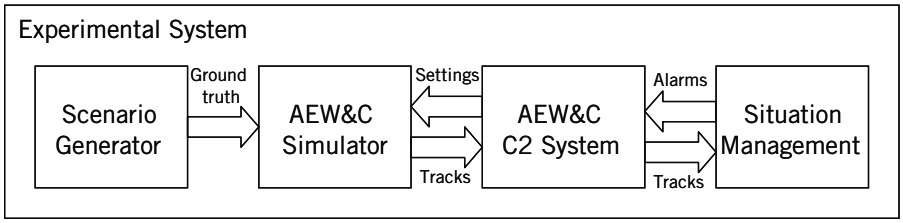


Figure 4.10: Overview of the components in the experimental system.

4.2.3 Experimental system

The experiments were conducted using an existing AEW&C system called Erieye [6], combined with the situation management system described in Section 4.2.2. The data used in the experiments was derived from a high-fidelity radar simulator that includes the same data processing components as the operative system. The application area used in the experiments is the maritime area; airborne objects were not considered. This is mainly because the knowledge-driven agents are constructed for finding maritime threats.

4.2.3.1 System Setup

Figure 4.10 shows the components of the experimental system. The Scenario Generator and the AEW&C Simulator come from a class-room trainer system, while the visualisation and operator interaction is done by the real AEW&C Command & Control (C2) system. The situation management system is connected to the C2 system.

The task of the scenario generator is to generate high-fidelity ground truth data of vessels. This data is fed into the simulator which based on the operator settings, generates tracks for all detectable objects in the ground truth data. The tracks are presented to the operator on the C2 system. The situation management system uses a common track interface to retrieve the current set of tracks from the C2 system every second. The track interface is also used to update the affiliation of specific tracks. This is the way that the situation management system visualises alarms to the operator. Possible values of the affiliation are Friend, Neutral, Suspect and Hostile. This limits the amount of situational information that can be transferred to the operator. To demonstrate all the capabilities of the situation management system, a small demonstration GUI was developed. This GUI can show vessels tagged as Friend, Anomaly, Suspect Raid, Hostile Raid, Suspect Pursuer and Suspect Smuggler. The GUI is shown in Figure 4.11.

4.2.3.2 Scenarios

There were two main scenarios used in the experiments. The first was a normal scenario without any illegal or abnormal activities, and included about 500 vessels travelling along the Swedish west coast. The left part of Figure 4.11 shows a snapshot of the scenario. The vessels in the scenario were a mix of large and medium sized ships as well as small boats. In the second scenario, a number of vessels were added. The behaviours of these included a number of smuggler, raid and pursue situations. In addition, the scenarios were approximately one hour long and included a total of 12 anomalies.

The track data derived from the C2 system includes the latitude and longitude of the vessel, time-stamp, velocity, heading and affiliation.

4.2.3.3 Operator Involvement

The operator can use the C2 to choose what information to see. For example, it might be interesting to see vessels tagged as hostile, while those tagged as suspect could be of no interest. The operator can also manually change the affiliation of a vessel, if, for example, he or she wants to reclassify a reported raid as normal.

4.2.4 Results and Discussion

The first scenario with no illegal activity was used as training data to build the normal model used by the anomaly agent. The second scenario ran in real-time to evaluate how well the agents could detect the raid, smuggler and pursue situations. This resulted in the agents actually finding all of the instances of the interesting situations. Moreover, the anomaly agent detected the situation by classifying tracks involved in the situations as anomalies prior to the detections from the knowledge-based analyser agents. Figure 4.11 shows two snapshots from the experimental GUI that demonstrates the advantage of having an automatic situation management system capable of helping the operator to focus on the interesting parts of the complete situation picture.

Figure 4.11 shows all the vessels without any specific tagging from the situation management system, while Figure 4.12 displays all the vessels tagged as Anomaly, Suspect Raid, Hostile Raid, Suspect Pursuer or Suspect Smuggler. This can help the operator focus his attention on some interesting subsets of the complete situation picture.

The scenario used for the experiments included a number of illegal activities that were quite easy for the agents to find. In a real scenario, the behaviour of the vessels is probably much more random and with a greater variance. This means that the agents should have a much harder time finding the specified behaviours. Through the design of the system, it is easy to include real radar

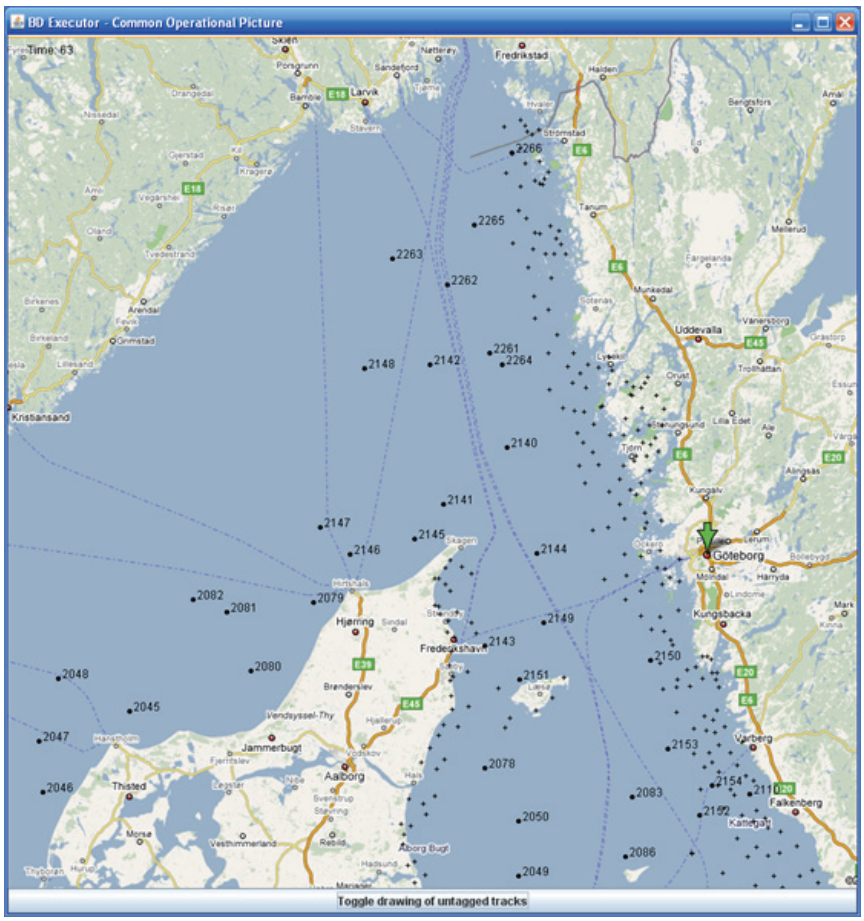


Figure 4.11: Demonstration GUI showing an unfiltered situation picture.

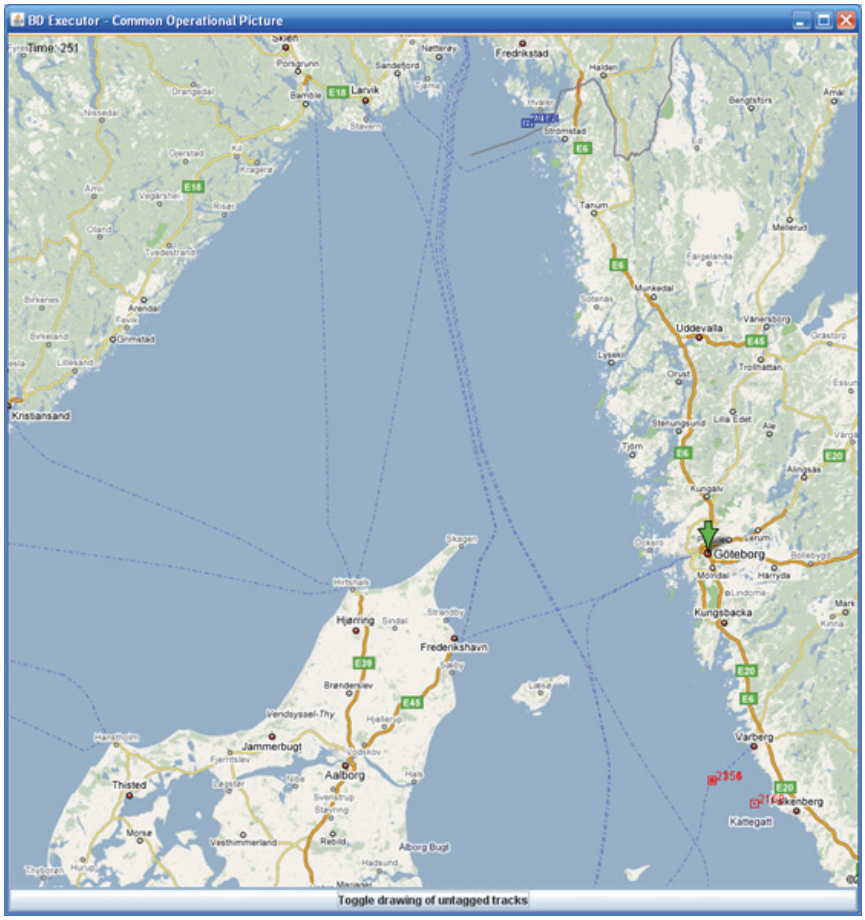


Figure 4.12: Demonstration GUI showing a filtered situation picture based on information from the situation management system (lower).

and AIS data for a more thorough evaluation of the actual performance of the system.

When dealing with data-driven approaches, it is important to update the normal model to reflect what is normal in the domain where the system is to be used. In this study one normal model was built and used for all the experiments. This is due to the fact that the domain is constant. If the domain should be subjected to change, for example, if a new port were to open somewhere along the coast, the normal model has to be revised to reflect the new behaviours of objecst in the domain. This in an important aspect when dealing with real-world problems and should be evaluated in future experiments.

4.2.5 Summary

The result of the evaluation supports the claim that the operator's workload can be reduced by filtering out interesting situations. Although this is only preliminary work with quite simple scenario, this type of capability is vital for detecting new asymmetrical threats from piracy or terrorists. The most interesting result was the data-driven anomaly detection system's ability to provide an early warning by finding vessels that were about to implement one of the scenarios considered interesting, and which should have been detected by the knowledge-driven detection agents.

In order to obtain a better evaluation of the situation management system, it should be assessed with real data from the radars and with AIS (Automatic Identification System) data [54]. Moreover, the integration with the C2 system should be studied more closely. What kind of interactions does the operator need in order to obtain as much support as possible from the situation management system? How can the cause of an alarm raised by the anomaly detector or one of the knowledge-based analyser agents be explained to the operator? Should some kind of story telling scheme be used to tell the operator how the tagged vessels behaved during the last moments before the alarm? Another aspect of the operator interaction that should be investigated is how to set thresholds and other parameters in the agents. Is it possible for the surveillance operator to set parameters or should this only be possible for specially trained technicians?

Chapter 5
Extending the State-Based Anomaly Detection Method

This chapter addresses research objectives two, three and four. The content of the chapter is largely based on the following publications[1]: Brax and Niklasson [27], Brax et al. [30]. The main contributions of the chapter are:

- An extension to the State-Based Anomaly Detection (SBAD) method for detecting anomalous state transitions and anomalous stops as well as a new method for detecting temporal anomalies with a fusion scheme for fusing the result from multiple detectors.
- An evaluation of the SBAD method on a dataset based on GPS data from trucks and commuters.
- A performance comparison between SBAD and Gaussian Mixture Models on indoor video surveillance data.

5.1 Experiments in the Area of Land Transportation

This section describes an extension to the SBAD method and how it can be used to find anomalies in the application area of land transportation. In this area, the objects that are surveyed mainly consist of vehicles travelling along roads, e.g., cars, trucks and busses. The experiments in this section were conducted together with Volvo Technology AB as part of a transportation security project at Security Arena, Lindholmen Science Park, in Gothenburg.

[1]Design, implementation and evaluation of the momentary track features approach in Section 5.2, was based on the co-author, Rikard Laxhammar's contribution to the Brax et al. [30] paper.

5.1.1 Introduction

In recent years, the rapid development of low cost GPS receivers and mobile broadband networks has enabled fleet managers to equip all their trucks with GPS tracking equipment. This transforms fleet management systems to advanced command and control (C2) systems, similar to the ones used by military commanders. Huge amounts of information can be gathered and must be analysed in real-time. If the operator is presented with too much information he or she can experience information overflow [133], and the ability to make the correct decisions may decrease. We have previously argued that the manual analysis of surveillance data can give good results under optimal conditions (few sensors, sparse scenes and small areas of interest). However, in many cases, limitations inherent to human analysis, such as boredom, fatigue, inconsistent analysis, lack of experience and so on, result in a low level of probability that the operator will find the interesting information in the data.

The traditional solution to the information overflow problem in the military domain is to add more people to analyse the information. This is possible to some extent but much research [64, 75, 76, 96, 108, 116, 119] now focuses on how to automatically analyse situations and filter the information before it is presented to the operator. In the transportation, domain it is not economically feasible to add more operators to the system. Therefore, the amount of information presented to the operator in a fleet management system must be reduced to be of any use. If not, the most common use of the data is an after-action review for forensic purposes, in which case, the fleet commanders miss the opportunity to be proactive.

We have previously shown the advantages of using data-driven anomaly detection to automatically analyse and filter information from closed-circuit television (CCTV) cameras and airborne radars (see Chapter 4). The CCTV cameras were used for close area surveillance and the airborne radar for maritime surveillance. The main task for the decision support system in these domains is the same as in the transportation domain, i.e., to help the operator focus on the most important pieces of the available information. In other words, the task is to efficiently filter the information based on its importance to the operator. It is assumed that the behaviour of the objects surveyed determines their importance and that unusual behaviour is more important than normal behaviour. One approach for this kind of filtering is data-driven anomaly detection, where recorded data is used to automatically learn what characterises normal behaviour. The result is usually a statistical model called the "normal model" which describes the normal behaviour of the objects of the domain of interest. The normal model can be used in real-time to classify the behaviour of new objects surveyed by the system as *normal* or *anomalous*.

There are several differences in the data from land transportation compared to the data from video and maritime surveillance. The available data from the land transportation area only covers a few of the objects and there is no infor-

mation about the rest of the objects. On the other hand, the uncertainty in the available information is much lower in the transportation data. This is mainly because the resolution of the position information is much higher using GPS transceivers compared to using airborne radars or CCTV cameras. In addition, the behaviour of the surveyed objects differs considerably between the areas. Maritime vessels can move with many degrees of freedom, but in the transportation domain, vehicles are usually bound to the road network and, hence, there is less variation in the normal data. This difference can make it easier to find anomalies such as objects deviating from normal routes.

The aim of this section is to evaluate how well the representation and normalcy modelling, proposed in Chapter 4 perform when used to find anomalies in land transportation data. Smuggling and accidents are interesting anomalies and data from GPS transceivers provides the input to the anomaly detection system.

5.1.2 Methodology

This section describes the methodology used in the experiments.

5.1.2.1 Experimental System

To evaluate wheater the State-based anomaly detection algorithm performs well in the area of land transportation, an experimental system was developed on the basis of the suggestions in [101]. The system was used for a number of different experiments including driver identification, electronic cargo seals, information system integration and detection of anomalies. In this experiment, we focus on the anomaly detection experiments conducted as part of the larger study.

The experimental system consists of three subsystems: i) a truck with a GPS receiver and mobile broadband capable of sending GPS data to a fleet management system, ii) a fleet management system that handles all the trucks in the fleet, work orders, cargo manifests, events and so forth and iii) an anomaly detection system that analyses data from the fleet management system and generates anomaly alarms which are sent back to the fleet management system.

In order to collect data for the experiments, one truck was used for one month, delivering goods as usual while connected to the fleet management and anomaly detection system.

The goal of the experiments was to demonstrate that it is possible to detect anomalies such as smuggling and accidents. Smuggling can be carried out in a number of ways, but in this project, we were interested in smuggling scenarios where partially loaded stop somewhere between the pickup and delivery address to load additional goods without the original contractor's knowledge. As with smuggling, accidents can occur in a number of different ways, but we were interested in those that will cause a truck to stop or move slowly for a while.

Table 5.1: Dataset statistics.

	Training dataset		Test dataset	
	Number of tracks	Number of observations	Number of tracks	Number of observations
Truck	14	15612	5	4459
Commuter	37	19889	22 (11 normal, 11 anomalous)	10805

Since this type of behaviour can also be caused by traffic congestion, the system must be able to learn the difference between an accident and traffic congestion.

5.1.2.2 Dataset Description

The main dataset used in the experiments comprised GPS data from a truck. However, there were some problems with this dataset. The start of data collection was delayed and, when the data collection was completed, the resulting dataset was much smaller than expected, which was mainly due to a dramatic decrease in transports at the end of 2008. As a result of these problems, a second dataset was created for testing and tuning the anomaly detection algorithm. This dataset also included GPS data, but was derived from a smaller area and from a number of commuters equipped with GPS receivers. The dataset was collected in real-time, but all processing was done off-line. Both datasets were constructed from a number of GPS-readings with the following features: Time-stamp, Latitude and Longitude coordinates and Altitude. Given these features, the velocity and course were calculated. The commuter dataset included one GPS-reading every second and the truck dataset included one every fifth second.

One problem with data-driven anomaly detection is that the data is unlabelled, i.e. each observation is not labelled as normal or anomalous, which makes the performance evaluation much harder. To evaluate the extent of this problem and to fine tune the algorithms, the commuter dataset was manually labelled The labelling was done on the track level, i.e., if a track does not include any anomalous observations, the entire track is labelled as *normal*, but if it includes observations considered to be anomalous, the whole track is labelled as *anomalous*. Due to the small size of the commuter dataset, it was easy to manually label the tracks. If the SBAD method were to be used on a larger dataset, it cannot be assumed that all the data can be manually labelled. The labels in the commuter dataset is only used for evaluation and not for learning the normal model.

Each of the datasets was divided into two subsets; one called the training dataset, which was used to build the normal model, and the other called test dataset, which was used to evaluate the normal model's classification performance. A good normal model should make the correct classification of the data in the test dataset. Table 5.1 shows some statistics regarding the number of tracks and observations in each dataset.

The commuter test dataset included both normal tracks without any anomalies and anomalous tracks with inserted anomalies. There were a number of different classes of inserted anomalies. The first class of anomalies was a deviation from the normal route, which incorporated a complete stop and then continuing on the normal route. This class of anomalies simulates a smuggling scenario, and is an anomaly described by Nyqvist and Bergsten [101]. Another class of anomalies is a complete stop, when travelling on a normal route, and then continuing on the normal route after a period of time. This simulates a traffic accident scenario. The dataset also included a short-cut scenario incorporating a deviation from the normal route for a period of time and two instances of snow-storm anomalies that contained reduced velocity in the normal routes.

5.1.2.3 Anomaly Detection Method

There are a number of important aspects to consider, when developing a data driven anomaly detection method. First and foremost, is to find a suitable representation of the data, which captures all the features of interest to us in the available data. For example, if we do not include the velocity of a truck, we cannot expect the system to find anomalies connected to the velocity. Another important aspect of the representation is whether it supports the incorporation of contextual information, such as GIS data, traffic statistics, time of day, timetables and to on. In this experiment we have used the State-based anomaly detection algorithm introduced in Chapter 4. The algorithm uses a discrete state-based representation to model both context information and real-time data from sensors. The basic model is similar to multi-dimensional histograms [100], but has been adapted to handle the incorporation of domain expert knowledge regarding what can be expected in the area of interest.

The anomaly detection problem differs from ordinary classification problems, since the training data used to build the classification models is unlabelled, i.e., all the training data is assumed to be of the same class, the normal class. This makes is hard to use methods that assume labelled data with multiple classes, i.e., decision trees, support vector machines, rule-based classifiers and neural networks based on supervised learning [132]. Instead, clustering methods are usually employed to find clusters of normal observations. The state-based method used in this experiment is a kind of clustering approach where all possible clusters are defined in advance and the data supplies information about the number of observations that ends up in each cluster.

The state-based representation is built on a number of state classes. A state class is a discretisation of a feature, for example, we use the state class *CourseState*, which is a discretisation of the course feature. A state class has a number of states, in the example above the *CourseState* might have four different states: north, south, east and west. In order to represent the features from the GPS-receiver, we used four different state classes:

- *CourseState* with eight different states: north, north west, west, etc.
- *PositionState* with a 20x20 grid placed around the area of interest.
- *SpeedState* with four states: stopped, slow, medium and fast.
- *TimeState* with four states: morning, noon, afternoon and night.

The reason for using these four state classes is that they capture all the important features of the available information. However, the altitude information from the GPS is not included in the model. The course information is included in the model to make it possible to distinguish between objects travelling in different directions on a road.

The upper and lower limit of each state is set by a domain expert with respect to what can be expected in the data from the specific domain. For example, the *SpeedState* levels for trucks are 5, 40 and 80 km/h, i.e., at a speed between 0 and 5 km/h the trucks velocity is classified as *SpeedState=Stopped*, between 5 and 40 km/h *SpeedState=Slow* and so forth. These levels could also be set with respect to the distribution in the data, using automatic discretisation (such as proposed by Liu et al. [94]), which is, however, outside the scope of the experiment. This is because operators should be able to interpret the output from the anomaly detection and, for an operator, a speed classification presented as *Fast* is more intuitive than *SpeedClass12*, which could be the case, if the discretisation was to be done automatically.

The four state classes above are called atomic states classes. We also use composite state classes, which are built by combining atomic state classes. In this experiment, we only used one composite state class, which was built from one instance of each of the four atomic state classes. It is assumed that all four atomic states are correlated and should be modelled accordingly. An example of a composite state class is the following: *CompositeState_A = (CourseState = north, PositionState = row12_col5, SpeedState = Stopped, TimeState = morning)*. CompositeState_A is the result of a GPS data entry with the course north (±22.5 degrees), a latitude and longitude position ending up in row number 12, column number 5 in the position grid, a velocity between 0 and 5 km/h and a time-stamp between 05.00 and 10.00 A.M.

When a suitable representation of the data has been accomplished, the next step is to create a normal model. In this experiment, we used the same kind of normal model as in Chapter 4. The normal model is built by collecting statistics

about the relative frequency of composite states in the training dataset. Building the normal model is done according to the following steps:

1. Take one observation from a track in training data and create atomic and composite states based on the features of the observation.
2. Add one instance to the number of instances for the specific composite state.
3. Repeat for all the observations of all tracks in the training dataset.
4. Divide the number of instances for each composite state with the total number of instances to obtain the relative frequency or probability of each composite state occurrence being normal.

We have previously shown that this kind of normal model can be used to detect anomalies where one or several of the atomic states occur rarely. A threshold called T_{Occ} is used to change the sensitivity of the detector. The threshold can be set to 0 to classify all composite states not present in the normal model as anomalous or to higher than 0 to allow for both new and rare composite states to be classified as anomalous.

In the commuter dataset, it is common to travel north at low to medium speed during the morning in a specific geographic area. It is also common to travel south at low to medium speed in the afternoon in the same area. However, if we observe a commuter travelling south in the morning, the normal model will classify this observation as anomalous. We call this kind of anomaly *State Occurrence Anomaly*. The smuggling anomaly could potentially be found in the same way, because stopping to load additional gods in an area where a truck usually does not stop constitutes a state occurrence anomaly.

There are however, some problems when only state occurrences are used in a normal model. Imagine a traffic accident where the observed vehicle suddenly reduced its velocity. This can be detected as a rare state occurrence, but it is possible that the lower velocity in that area is normal due to earlier traffic congestions. However, the fact that we went from a composite state with high velocity to a composite state with low velocity might not be normal. To capture this in the normal model, we need to extend it with information about composite state transitions. While this approach is similar to the one proposed by Bui et al. [36], the difference is that this approach uses more features in the states (Bui et al. only use position), however, the state transition model is simpler and does not use HMMs. This makes the transition probabilities easier to calculate, but does not account for the time dependences in the data in the same way as HMMs do. The state transition analysis is implemented by analysing the sequence of composite states for each track in the training data and counting the number of state transitions for each pair of composite states. For example, Composite_A →Composite_B, Composite_A →Composite_B, Composite_B →Composite_C and so forth. If the number of state transitions for

each pair of states is divided by the total number of state transitions we obtain a probability value for each state transition. This value can be used to find *State Transition Anomalies*, defined as state transitions with a probability lower than a threshold T_{Trans}. This threshold can be set to 0, if we only want new, previously unseen state transitions to be classified as anomalous, or above 0, if we want the system to be more sensitive to state transition anomalies.

The *transitions* between composite states $\langle \mathbf{y}_i, \mathbf{y}_j \rangle \in \Omega_{\mathbf{Y}\times\mathbf{Y}}$ can be formalised as:

$$p_{Tr}\left(\langle \mathbf{y}_i, \mathbf{y}_j \rangle\right) \triangleq \frac{n\left(\langle \mathbf{y}_i, \mathbf{y}_j \rangle\right)}{\sum_{\langle \mathbf{y}_i, \mathbf{y}_j \rangle \in \Omega_{\mathbf{Y}\times\mathbf{Y}}} n\left(\langle \mathbf{y}_i, \mathbf{y}_j \rangle\right)}, \tag{5.1}$$

where $n\left(\langle \mathbf{y}_i, \mathbf{y}_j \rangle\right)$ denotes the number of transitions $\langle \mathbf{y}_i, \mathbf{y}_j \rangle$ over all tracks and where i and j denote two consecutive time steps from the same track.

State transitions between the same state, such as Composite_A $\rightarrow$ Composite_A, are a special case of state transitions that is handled separately. Consider the truck that stops for a while to load illegal gods, which would result in a number of state transitions between the same states. It is important to detect anomalies that include an object which stays in the same state too long. Therefore, another set of information which models the normal length of say in the same state is added to the normal model. For example, it might be normal to stay in Composite_A for one to five time steps, which could happen when a truck is waiting at a gate. However, if the truck stays in Composite_A for six or more time steps, truck should be classified as an anomaly. We call this a *Same State Anomaly*, and an observation is classified as a same state anomaly, if it stays in the same composite state longer than the maximum number of states defined in the normal model.

Assume that an object has been in state $y \in \Omega_{\mathbf{Y}}$, for t time steps. Then, we can formulate a model $p_{Sa,y}\left(t\right)$ that expresses how normal it is for an object to stay at least $t \in \Omega_t^y$ time steps in a state y:

$$p_{Sa,y}\left(t\right) \triangleq \begin{cases} 1 & n_y\left(t\right) > 0 \\ & \\ 0 & n_y\left(t\right) = 0 \end{cases}, \tag{5.2}$$

where $n_y\left(t\right)$ denotes the number of objects in the training dataset that stayed for t time steps in composite state y. Note that the probability is binary; either zero, indicating that it is an anomaly, or one indicating that it is normal.

The last aspect to consider is what is needed to classify an object as anomalous. It is usually not enough that one observation of the object is classified as anomalous, because one anomalous observation can occur from noise in the system. To obtain a more robust system, we argue that it is important to look at multiple observations, instead of just the last one. If the observations for a vehicle are classified as anomalous for more than n time steps, the vehicle can

be classified as anomalous. Looking at multiple observations can be done in two different ways, during the whole track or for an interval shorter than the whole track. In this experiment, the latter method was used to avoid that the length of the track affected the probability of finding an anomaly. When using an interval, it is possible to detect vehicles that go from normal to anomalous and then back to normal again. The interval can be implemented by a sliding window, and the size of the sliding window is defined by a value called n_{ws}. The use of multiple observations for classifing a track is similar to the cumulative detection approach by Kraiman et al. [85].

The next problem that arises is how to fuse the output from the three classifiers described above. Should their classification be added or multiplied, have the same or different weights? In this experiment, we choose to simply multiply the output from each classifier with a weight and add them together. After an empirical study of the detection performance with different fusion schemes, the following weights were found: $2/k$ for the occurrence anomaly classifier and $1/k$ for the transition and same state anomaly classifiers, where k is the number of classifiers.

The classifiers used in this experiment give the binary output zero, if the observation is classified as normal, and one, if it is classified as anomalous. For each time step, the n_{ws} last observations are classified by each of the three classifiers. The anomaly value for each classifier and for the fused degree of anomaly will be in the interval$[0, n_{ws}]$. The fused degree of anomaly is denoted DOA_{Tot} and is used together with a threshold T_{al} to decide whether a vehicle should be classified as normal or anomalous. DOA_{Tot} is calculated as follows:

$$DOA_{Tot} = \sum_j \left(\lambda_j \sum_{i=t}^{t+N} C_j \left(\mathbf{z}_i \right) \right), \tag{5.3}$$

where N is the window size (also known as n_{ws}), C_j is a binary anomaly detector, $\mathbf{z}_i$ is an observation, λ_j is the weight for the C_j detector. Note that $\mathbf{z}_i$ might be a single composite state in case of the occurence and same state classifier, or two subsequent compoiste states, in the case of the transition classifier. A track is classified as anomalous if:

$$DOA_{Tot} \geq T_{al}, \tag{5.4}$$

where T_{al} is the user specified detection threshold.

Table 5.2 shows an example of how DOA_{Tot} can be calculated for a track with n_{ws}set to five. A "0" denotes that the observation was classified as normal by the classifier and a "1" denotes that the observation was classified as anomalous. Equation 5.5 shows an example of DOA_{Tot} for the observations illustrated in Table 5.2. The weights λ are set to 0.5, 0.25 and 0.25 for the Occurrence, Transition and SameState detectors respectively, which results that DOA_{Tot} can be calculated as $0.5 * 3 + 0.25 * 2 + 0.25 * 1 = 2.25$. If we use a threshold T_{al} set to 2 the observation at time C_t would be classified as anomalous.

Table 5.2: An example of how the total degree of anomaly is calculated for a specific time step.

	t_{n-4}	t_{n-3}	t_{n-2}	t_{n-1}	t_n	Total number of anomalous observations for the 5 last time steps
Occurrence Anomaly	0	1	1	1	0	$DOA_{Occ} = 3$
Transition Anomaly	0	1	1	0	0	$DOA_{Tr} = 2$
Same State Anomaly	0	0	0	0	1	$DOA_{SameState} = 1$

$$DOA_{Tot} = 0.5 * DOA_{Occ} + 0.25 * DOA_{Tr} + 0.25 * DOA_{SameState} \quad (5.5)$$

5.1.2.4 Evaluation Method

To evaluate the performance of the anomaly detection algorithms, a modified version of the 10-fold cross validation [107] scheme was used. The purpose of the use of cross validation was to find a way of automatically setting the T_{al} threshold on the basis of the training data. The training dataset was randomly split into nine equally sized subsets based on complete tracks, i.e., no tracks were split. Eight of the subsets were used to build a normal model. The ninth subset was classified using the normal model, i.e., the value DOA_{Tot} was calculated for every observation in each track. The goal of this classification is to find a suitable value for T_{al} that can classify all normal tracks as normal and all anomalous tracks as anomalous. By using the highest value of DOA_{Tot} found in the classification of the tracks in the ninth subset as the threshold T_{al}, it can be assumed that no normal tracks in the training dataset will be classified as anomalous. This is repeated nine times using different subsets to find the threshold for each normal model. All nine normal models and their thresholds are then used in an ensemble for classifying the tracks in the test dataset. The total classification of the ensemble was devised by using the majority-voting scheme [98], where each member of the ensemble casts a single vote for the correct class. The class with most votes becomes the ensemble classification. To prevent the ensemble from reaching a tie, nine random members were used instead of ten.

A number of parameter settings for n_{ws} and the size of the position grid were empirically tested, using the evaluation method above. The outcome was

Table 5.3: Results of experiments with the commuter dataset.

True positives (Anomalies classified as anomalies)	10
False positives (Normal classified as anomalies)	1
True negative (Normal classified as Normal)	10
False negative (Anomalies classified as Normal)	1
Precision	90.9%
Recall	90.9%

that a window size of 20 and a position grid of 20 times 20 cells gave the best results. These settings were used for all the experiments.

5.1.3 Results

Due to the low number of tracks in the truck dataset, we chose to focus on the result from the commuter dataset. We also included the results from the truck dataset, even though they were inconclusive.

5.1.3.1 Commuter Dataset

The results of the anomaly detection in the commuter test dataset can be found in Table 5.3. The 22 tracks in the test dataset were manually labelled as normal or anomalous. Half of the tracks in the dataset were normal and half were anomalous. Ten of the normal tracks were correctly classified by the ensemble, while one was incorrectly classified. The ensemble also classified ten of the eleven anomalous tracks correctly. All the smuggling anomalies, as well as the shortcut anomaly and full stop anomaly were correctly classified. The anomalies hardest to find were the snowstorm anomalies, where one of two was correctly classified.

The results are encouraging; the classifiers used in the ensemble were able to capture most of the normal behaviours in the training dataset and classify the test dataset with high precision and recall. The dataset is, however, too small to draw any overall conclusions about the method's general performance in the land transportation domain.

5.1.3.2 Truck Dataset

The evaluation of the truck dataset was done differently compared to the commuter dataset. The small number of tracks and the fact that the test dataset was unlabelled made it impossible to obtain reasonable results in the cross validation evaluation. Instead, the dataset was evaluated by using the first three weeks of data, i.e., the training dataset, to build a normal model. This model

was then used to classify the tracks in the test dataset. The resulting classification was manually analysed.

The anomaly detection algorithm found anomalies in three of the five tracks. In the first anomalous track, the truck used another route when leaving the main distribution centre. It also, at noon, used a route from the distribution centre to the pick-up location usually used in the morning or at night. The second anomalous track included the truck picking up goods at an unknown location as well as driving a new route to the ordinary pick-up location. The third anomalous truck included a large geographical jump due to a malfunction in the on-board computer which caused it to reboot, as well as travelling at noon on a route usually travelled in the morning. None of the anomalies above were reported as "real" ones by the driver of the truck, nevertheless they could be of great importance for the companies providing the shipping services. If more training data were available, some of the routes detected as anomalies would possibly be included in the normal model and, hence, not be subjected to anomaly alarms.

5.1.4 Summary and Discussion

In this section, the initial results from a supply chain security project are presented. The project's aim was to solve one part of the supply chain management problem, the monitoring of transported goods. One truck was equipped with a GPS transceiver and reported its status during a month of regular operations and a number of commuters recorded their daily routes with GPS receivers. A data-driven anomaly detection method was evaluated using the collected datasets from the truck and the commuters. The latter was a more controlled dataset including both normal tracks and deliberate anomalies. This dataset was manually labelled, i.e., the tracks in the dataset were manually classified as normal or anomalous.

The results from the commuter dataset indicate that it is possible to detect anomalies such as smuggling and accidents, with a low false alarm rate. Information regarding anomalies can be used as decision support for an operator. Moreover, the results of an anomaly detection system, can make it possible to increase the long term efficiency of the transportation system, by increasing security and reducing the risk of interruptions in the supply chain.

The results from the truck dataset are inconclusive, due to a smaller dataset than expected and low variation in the data, i.e., there were no real anomalies in the transports. Nevertheless, a number of potentially interesting anomalies were found.

Another contribution of this experiment is a method for automatically setting the thresholds for classifying unlabelled data. This method is based on the n-cross validation scheme together with a voting ensemble.

What remains to be done is to evaluate how different methods of fusing the output from the three anomaly classifiers (occurrence, transition, same state)

as well as different ways of building the ensemble. If the anomaly detection system were to be integrated as a decision support component in an operative fleet management system, the user interaction should be developed further. Another important part in an operative fleet management system is to include knowledge-based anomaly detection, i.e., the ability to capture the knowledge of domain experts in order to detect both known, describable anomalies as well as new, previously unseen ones.

5.2 Experiments in the Indoor Video Surveillance Domain

This section elaborates on how the SBAD method can be used to detect anomalies in an indoor video surveillance scenario. The work was part of a Research and Development project at Saab called RAHN (Real-time Anomaly detection in Herogenous Networks). The project was a cooperative endeavour between Saab and the LFV Group (Luftfartsverket) at Landvetter Airport. Saab provided all the technical systems and LFV Group provided a scenario location and access to security experts.

5.2.1 Introduction

In the previously presented experiments using an intelligent surveillance system (Chapter 4 and [29]), a video tracker was used to extract information from two CCTV cameras. The information was used for anomaly detection, and the detection algorithm was able to detect a number of different anomalies. However, the environment used in the experiments was not very crowded; only a few objects were present in the scene at any given time and their motion patterns were mainly regular.

This section extends the previous work by looking at a much more crowded environment, in combination with advanced tracking and anomaly detection. The environment constitutes the international departure hall of an airport. Two different methods of anomaly detection (SBAD and Gaussian Mixture Models) are compared and evaluated individually as well as collectively, i.e., with their output fused. The hypothesis is that by combining two different methods of anomaly detection, we should be able to find more anomalies than when using just one method.

5.2.2 Methodology

An experimentation platform was built to evaluate whether the classification performance of an anomaly detection system could be increased by running multiple anomaly detection algorithms in parallel and fusing their output. The

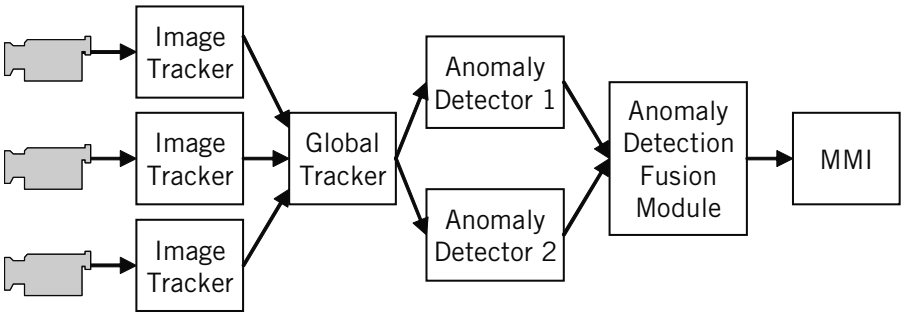


Figure 5.1: System architecture for the experimental system.

video data for it was recorded at the international departure hall of a major Swedish airport.

5.2.2.1 Platform Architecture

The experimental platform used for all the experiments is based on the one described in [28, 29], with some exceptions. Figure 5.1 shows the platform's schematic design. The video tracking subsystem consists of three single CCTV cameras with software for extracting tracking information connected to each camera. These image trackers produce *local tracks* described in spherical coordinates relative to each camera. The local tracks are sent to a global tracker which integrates the local tracks to produce *global tracks* in a Cartesian coordinate system with a fixed origin. The anomaly detection subsystem consists of two anomaly detection modules which analyse global tracks and add an estimate of the degree of anomaly to each track. The output from the anomaly detection modules is then fused by the Anomaly Detection Fusion Module to obtain a single estimate of the degree of anomaly for each track. Finally, the anomaly classification is presented to an operator on a Man Machine Interface (MMI), together with the video and tracking information.

All processing is done on-line in real-time, with the exception of the video, which is recorded in advance. Instead of using live video from CCTV cameras, pre-recorded video is played back to the image tracker. This provides a controlled environment in which to analyse and compare the performance of the individual and combined anomaly detectors objectively. The functionality of the demonstration platform is exactly the same regardless of whether live or pre-recorded video is used.

One of the new features of the video tracking subsystem compared to the one used in [28, 29], is the ability to track objects between two, non-overlapping cameras. This is an important feature that will potentially have considerable

Table 5.4: Statistics for the dataset used in the experiments.

	Day 1	Day 2	Day 3
Total number of tracks considered normal	18538	29103	36338 (13995 used in test dataset)
Number of anomalous tracks	0	0	281 (used in test dataset)

impact on the anomaly detection performance, when dealing with anomalies spread over a wide area that may lack complete camera coverage.

5.2.2.2 Dataset

The video was recorded between 7 A.M. and 7 P.M. for three consecutive days. The data from the first and second day was used as training data and implicitly assumed to reflect the normal behaviour of people at the airport. During the third day, a number of deliberate anomalies were recorded together with the normal activity. These anomalies were manually classified; all other observations were considered to be normal. Table 5.4 shows how many normal and anomalous tracks were recorded during each of the three days. A subset of Day Three's tracks was used as test data for the experiments. This subset included 281 anomalous tracks as well as 13995 normal tracks that were present in the same data.

Even though all the tracks during day one and day two are assumed to be normal, it is possible they contain some tracks that could be classified as anomalous. This could potentially raise a problem. However, so long as the number of anomalous tracks is sufficiently lower than the number of normal tracks, the anomalous tracks should not have a significant negative impact on the classification performance of the anomaly detection algorithms. The focus in the experiments was on the performance of the detectors and not on the quality of the training data. High performance would suggest that the impact of any potential anomalies in the training data is low. Low performance would suggest that the detection algorithms have poor performance or that the impact of potential anomalies in the training data is high.

Figure 5.2 shows the layout of the departure hall and the approximate positions and field-of-view of the cameras. Two check-in counters and two staircases/escalators to the security check area are also depicted.

The global tracker outputs tracks with the following features:

- ID.
- Time-stamp.

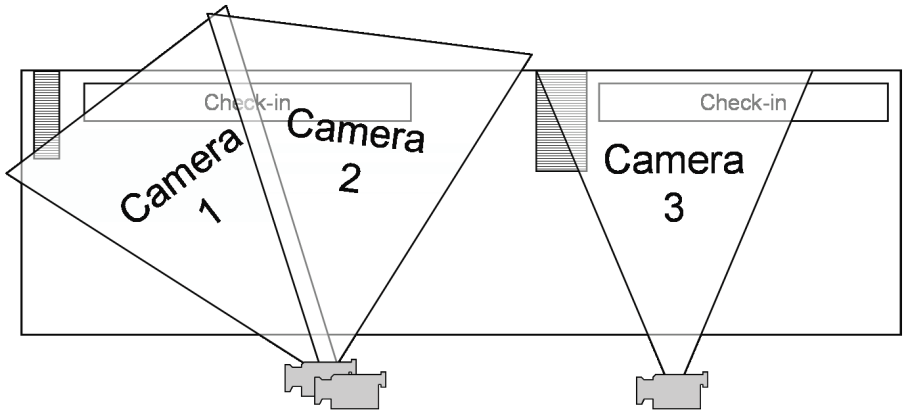


Figure 5.2: Camera setup at the demonstration site.

- X, Y and Z coordinates for the object.
- X, Y and Z velocity for the object.
- Width and height of the object.

The refresh rate provided by the trackers is 4 Hz for the global tracker and 15 Hz for the local tracker. The video frame rate is 15 Hz.

During the third day, a total of twelve different types of anomalies were recorded; see Table 5.5. A number of variants were recorded for some anomalies. In total, 25 different anomalies were recorded and manually labelled. Some of the 25 anomalies were recorded at more than one time, e.g., in the morning and then in the afternoon. A total of 40 unique anomalies were recorded for a total of 281 anomalous tracks.

Some of these anomalies can be detected by using a single camera, e.g., the abandoned bag (1), while other anomalies, such as the loitering person (3), require tracking over multiple cameras which may have non-overlapping fields of view.

5.2.2.3 Algorithms

Anomaly detector 1: State-Based Anomaly Detection The first detection algorithm used is *state-based anomaly detection*, which we have used previously for finding anomalies in video surveillance data [28, 29]. The algorithm employs discrete state-based representation for both contextual information and real-time data from sensors; the representation is similar to multi-dimensional histograms that have traditionally been used for query optimisation in databases [100]. The main difference to multi-dimensional histograms used in Nitin et al.

Table 5.5: A list of the types of anomalous situations used in the experiments.

No	Description
1	Someone leaves a bag and walks away (4 variants).
2	A bag is picked up by another individual than its owner (2 variants).
3	A loitering person walks around in the hall for a long time without checking in (2 variants).
4	A person runs through the hall.
5	A person moves in the wrong direction on an escalator or staircase (2 variants).
6	A person runs from the check-in counter towards the security check (2 variants).
7	A person walks around in unusual places (2 variants).
8	A person stands still where people usually do not (2 variants).
9	3-4 persons run in different directions from one starting point (2 variants).
10	3-4 persons form a group at an unusual location (3 variants).
11	Pick pocket, bag theft: One person walks up, takes something, and hands it on to someone else (2 variants).
12	A person walks from group to group.

[100] is that the SBAD method uses expert knowledge to set the boundaries of the bins instead of generating them from the data.

The state-based representation is built on a number of *state classes*. A state class is a discretisation of a feature, e.g., course (or heading) and encompasses a number of possible states. In this experiment, the following state classes were used:

- CourseState, with nine different states: eight compass points (north, north-west, west, etc.) plus undefined
- PositionState, with a 10x5 grid placed around the area of interest.
- SpeedState, with four states: stopped, slow, medium and fast.
- RelationState, with five states: inFront, behind, left, right and undefined.
- EnvironmentalState, with six states: onFloor, onEscalator, onSeats, near-Machines, onStaircase and undefined.
- SizeState, with five states: small, medium, large, xlarge and undefined.
- TimeState, with four states: morning, lunch, afternoon and night.

The bounds for each state are set by a domain expert, based on what can be expected in the dataset given the domain. Thus, for example, the bounds for each state in EnvironmentState are set by comparing the position of the object to a map over the area.

The seven state classes above are called atomic states classes; there can also be composite state classes, which are built by combining atomic state classes. In this case we use only one composite state class, which is built from one instance of each of the seven atomic state classes. An example of a composite state class is:

> CompositeState_A = (CourseState=north, PositionState=row12_col5, SpeedState=Stopped, RelationState=inFront, EnvironmentState=onFloor, SizeState=small, TimeState=lunch).

CompositeState_A tracks an object with heading north (±22.5 degrees), x and y coordinates of (12,5) and velocity between 0 and 1 km/h. Another object lies in front of the object, the object is situated on the floor, it has a size under $0.3m^2$; it is observed during lunch hours.

After analysing the training data, we concluded that additional state classes were needed beyond those used in [29]. For example, it was very difficult to find a bag that was left unattended, because the representation scheme could not distinguish between a small bag and a large person. Therefore, the SizeState state class was added. It divides the training dataset into five classes based on height and width information from the video tracker. Another state class, called

TimeState, was added to handle daily fluctuations in the data. An observed behaviour is noted as occurring in the morning, at lunch, in the afternoon, or at night.

The basic approach for modelling normal activity is the same as in Section 4.1.2 and 5.1.2.3. Three different classifiers are used to find *occurrence*, *transition* and *same state* anomalies.

However, what is needed to classify a situation as anomalous? It is not enough that one or more of the three classifiers above classify a single observation as anomalous, since a single anomaly observation can be caused by noise in the system. To obtain a more robust classification, it is necessary to look at multiple observations instead of just one. If the observations of, e.g., a person are classified as anomalous for more than n time steps, the situation should be classified as anomalous.

The analysis of multiple observations can be done in two ways: over the entire track or over a shorter interval. The advantage of the latter is that it is possible to detect objects that go from normal to anomalous and then back to normal again. The interval is implemented as a sliding window of n time steps where the window size is defined by n_{ws}.

The last thing to consider is how to combine the three classifiers. Should they be added or multiplied; should they have the same or different weights? After a number of trials analysing different fusion strategies, we concluded that the occurrence classifier and the transition classifier should be weighted equally. On the other hand, although the SameState classifier was seldom triggered, when it was, it was often due to a genuine anomaly. Therefore, the SameState classifier is weighted to increase the total estimate of anomaly by n_{ws} over two, i.e., the same as if all an occurrence anomaly was detected in all the observations and a transition anomaly in half the observations within the sliding window. See [118] for a detailed review of classifier fusion methods.

For each time step, the last n_{ws} observations are classified according to each of the three classifiers. The value for each observation can be zero or one depending on whether an anomaly occurred. The total anomaly value for each classifier will be in the interval [0, n_{ws}]. The fused degree of anomaly is denoted by DOA_TOTAL. This is used together with threshold T_{al} to decide whether a situation should be classified as normal or anomalous. The total degree of anomaly is used by the Anomaly Detection Fusion Module. The thresholds n_{ws}, T_{al}, T_{Tr} and T_{Occ} are set on the basis of an empirical investigation of the number of false positives in the training dataset.

Anomaly detector 2: Anomaly Detection Based on Momentary Track Features
The second detection algorithm is based on momentary track features. It has previously been used for the detection of the anomalous behaviour of maritime vessels [87] and anomalous events in video surveillance data [28]. The algorithm uses a statistical model of momentary feature values in normal activity,

and then does anomaly detection based on the likelihood that newly observed feature values could have been generated by the statistical model. The basic feature model used in this study is an extension of the one used in [28]. Besides momentary kinematics (position and velocity vector), the accumulated age of each active track and the accumulated stationary time for stationary tracks were also used. The probability density function (PDF) for the normal values of these features is approximated by multivariate Gaussian mixture models (GMMs).

To reduce the complexity of the GMMs, the surveillance area is divided into a uniformly sized grid, where each cell models the local distributions of the features. Thus, for each cell, we have a four-dimensional GMM modelling the local position-velocity vector (capturing correlations between position, speed and heading of targets), a one-dimensional GMM modelling the local momentary track age and a third three-dimensional GMM correlating the accumulated time that objects have remained stationary in the position within each cell. During anomaly detection, the likelihood from each model is compared to a model specific threshold; if one or more likelihood is below the corresponding threshold, the observation is classified as anomalous. In this application, each individual observation is classified as being normal or anomalous. However, one may argue that it is more relevant to consider anomalous tracks rather than particular anomalous observations. Therefore, a track is classified as anomalous whenever one or more anomalous observations are associated to it.

Classifier Fusion

Studies have shown that a combination of classifiers can offer a significant performance increase for many classification problems [118, 86]. Therefore, we want to investigate whether the fused output from two anomaly detectors is better than the output from each of the detectors. To investigate this, we use a component called Anomaly Detection Fusion Module (ADFM). Inputs to the ADFM are system tracks from Anomaly Detector 1 and Anomaly Detector 2. The message from the anomaly detectors contains the same information as the global tracks derived from the global tracker. However, the anomaly detectors add more anomaly information about the tracks. The additional anomaly messages are: *Anomaly exist* (i.e., a byte telling if a track is anomalous or not), *Degree of anomaly* and a string, *Type of anomaly*, describing which feature or model has caused the anomaly. In the ADFM, these three messages are fused and the fused anomaly result, together with the other track information, is sent to the MMI.

In the current set up, the technique for mixing the anomaly messages from the two anomaly detectors is rather simplistic. The foundation of the fusion is the logical operator OR. This means that if one or both of the detectors discover an anomaly for a certain track, then the output from the ADFM for the same track will also be an anomaly. If only one of the detectors identifies an

anomaly, then the corresponding messages are just copied from the Anomaly Detector in question to the ADFM message. However, if both detectors discover an anomaly for the same track, then the value of the message, Degree of anomaly, is set to the highest value from the detectors. Furthermore, the string messages, Type of anomaly, from the detectors merged and are separated in the same string by a '+'-sign.

5.2.2.4 Evaluation

According to Dick and Brooks [48], the evaluation of surveillance systems is an important issue that is inherently difficult. This is due to the fact that most surveillance systems are very domain specific. Dick and Brooks argue that it is often best to make an extensive evaluation of a system in the environment in which it is intended to be used. However, there are many situations where this is not feasible. In these cases, quantitative performance measures, like Receiver Operating Characteristic (ROC) curves, can be used.

The aim of the evaluation in this experiment is to investigate whether a combination of two anomaly detectors achieves better classification performance than a single detector. The classification performance is measured by the number of true positives (TP) and false positives (FP).

A true positive, corresponds to a track tagged as anomalous, that is classified as anomalous by the classifier. A false positive, is a track tagged as normal, that is classified as anomalous by the classifier.

To evaluate the performance of the two anomaly detectors, data from the first two days is used to build the normal models. This data is called normal data and is assumed to be normal, i.e., the class of each track is "normal". The data from the third day is used as test data. This dataset includes both normal tracks and those with anomalies. The tracks with anomalies were manually labelled, i.e., their class label set to "anomalous", while the normal tracks were labelled "normal". The tracks labelled "normal" were not explicitly analysed and there is a possibility that some of them, in fact, constitute anomalies.

Both anomaly detectors have a number of thresholds that can be used to fine-tune the classification performance. In the experiments, three different threshold setups were used for each anomaly detector. The setups were tuned with respect to the classification performance; the goal was to use thresholds that resulted in approximately 1%, 3% and 7% alarms in the test dataset. It was decided that these thresholds were relevant, on the basis of discussions with subject matter experts. The approaches are compared by looking at the number of true positives for each level of false positives, i.e., at the 1% level, the number of anomalies detected by each detector separately and by both detectors together.

Table 5.6: Results of experiments with AD1 including four threshold setups corresponding to 1.1%, 3.3%, 7.2% and 9.9% alarms in the test dataset.

AD1: Threshold	1.1%	3.3%	7.2%	10.2%
True positives	12.5% (5 of 40)	32.5% (13 of 40)	45% (18 of 40)	62.5% (25 of 40)
False positives	1.1%	3.2%	7.0%	9.9%

5.2.3 Results

The results presented in this experiment originate from simulations where AD1 and AD2 were run separately. Therefore, we can only calculate the theoretical performance of both methods running together with ADFM. This is not a significant problem, due to the simple fusion scheme implemented in ADFM.

The results are compiled from summarizing how many of the 40 anomalies the detectors were able to find. An anomaly was considered to be found, if at least one of the tracks included in the anomaly was detected. While this is a lenient way of evaluating, it should be enough that only some parts of the anomalous situation are found to raise the attention of the operator.

The parameter settings for AD1 were decided by running a parameter search with different values for n_{ws}, T_{al}, T_{Tr} and T_{Occ}. After over 100 simulations, suitable threshold settings were found that resulted in 1.1%, 3.3% and 7.2% alarms in the test dataset. These values were as close as we could get to the 1%, 3% and 7% we aimed for. We also included results of settings corresponding to 10.2% alarms. The results using these four threshold setups can be seen in Table 5.6. The sum of the true and false positives is the same as the total number of alarms.

The parameters for AD2 were set with the intention of obtaining a combined 0.3%, 1% and 3% anomaly alarms from each of the three feature models. When combining the three models, the total number of alarms was 0.9%, 3.4% and 7.2% respectively. The results from AD2 are shown in Table 5.7.

The theoretical results of fusing the output from AD1 and AD2 are shown in Table 5.8. We used four different threshold setups for each anomaly detector; the first three resulted in approximately the same number of alarms in the test dataset. We also included one setup with the thresholds that found the most anomalies in each detector. The table should be interpreted as follows: for example, the second column, first row indicates that the 1.1% threshold setup was used for AD1 and the 0.9% was used for AD2. The second row includes the number of anomalies found when combining the output from the two anomaly detectors, i.e., 15% or 6 of the 40 in the test dataset. The third row shows the false positive interval. The lower bound comes from the detector

Table 5.7: Results of experiments with AD2 using three threshold setups corresponding to 0.9%, 3.4% and 7.2% alarms in the test dataset.

AD2: Threshold	0.9%	3.4%	7.2%
True positives	2.5% (1 of 40)	15% (6 of 40)	27.5% (11 of 40)
False positives	0.9%	3.3%	7.1%

Table 5.8: Results of fusing the output from AD1 and AD2.

AD1 + AD2	1.1% + 0.9%	3.3% + 3.4%	7.2% + 7.2%	10.2% + 7.2%
True positives	15% (6 of 40)	42.5% (17 of 40)	55% (22 of 40)	70% (28 of 40)
False positives	1.1-2%	3.3-6.5%	7.1-14.1%	9.9-17 %
True positive overlap	AD1: 5, AD2: 1, Both: 0	AD1: 11, AD2: 4, Both: 2	AD1: 11, AD2: 4, Both: 7	AD1: 17, AD2: 3, Both: 8

with the highest false positive number, i.e., AD1 with 1.1. The lower bound indicates that there is a complete overlap between the false positives for both the detectors. The upper bound comes from adding the false positives from both detectors together, i.e. 1.1% plus 0.9% equals 2%. This assumes that there is no overlap in the tracks detected as false positives. The fourth row shows the true positive overlap, i.e., five anomalies were only detected by AD1, one was only detected by AD2 and no anomalies were detected by both AD1 and AD2.

The results in Table 5.8 show that we can find more anomalies when two anomaly detectors are combined. However, it is not clear whether combining several detectors really is beneficial. It all depends on the false positive overlap. If the third threshold setup results in 7.1% false positives, it is better to use two detectors. However, if the setup results in 14.1% false positives, it is better to only use AD1 with the 10.2% threshold that finds three more anomalies with a 3.9 percentage point lower false positive value. We analysed the false positive overlap for the 3.3%+3.4% threshold setup in Table 5 and found that the overlap between AD1 and AD2 was only 1.1%, i.e., only 1.1% of all found false positives were found by both AD1 and AD2. This means that the fused number of false positives is closer to 6.5% than to 3.3%. Due to approximately the same overlap in the true positives between the four threshold setups, the overlap between the false positives should be approximately the same as well.

Therefore, we can assume that the fused number of false positives is close to the upper bound. This means that for the 7.2%+7.2% setup, it is better to only use AD1 with the 10.2% threshold. For the 1.1%+0.9% and 3.3%+3.4% setups, it is better to use the multiple detector approach.

5.2.4 Discussion

The results of the experiments are not as impressive as those for the video surveillance experiments in Chapter 4. The number of false alarms was much higher in this experiment compared to the ones in Chapter 4. This is quite natural due to the dramatic increase in complexity for the scene used in this experiment compared to the one used in Chapter 4. The increase in complexity was more demanding for the video tracking subsystem, which resulted in many tracks being separated in several sub-tracks. Consequently, the anomaly detectors find it difficult to detect anomalies based on stationary time, time in scene or time in same state. The unattended bag, anomaly 1 in Table 5.5, was supposed to be detected as an object remaining still for an abnormal amount of time, which only happened in two of eight instances, due to the fact that the bag changed identity a number of times. Even though the bag was stationary in the video images, the position and velocity reported by the tracker fluctuated. In an attempt to compensate for this in AD1, the CourseState was always set to *undefined* when the SpeedState was set to *Stopped*. Another reason for the low detection rate for unattended bags is that people usually put their bags in similar positions while they, for example, use the automatic check-in machines. It was also hard to find anomalies in the outer areas of the cameras' field-of-view, such as anomalies 7 and 8.

Another possible reason for the low number of found anomalies is that the scene includes a lot of people with many degrees of freedom. Their movement patterns were arbitrary, even if some people followed certain courses. This has an impact on the normalcy modelling in the anomaly detectors. The scenarios defined as anomalous in the experiments might be normal and included in the normal models and, hence, they will not be found by the detectors.

In this experiment, two completely different methods for anomaly detection were used. The approach suggested can exploit the specific advantages of each method. It can be argued that no single approach provides optimal performance for all possible anomaly detection problems, hence, a combination of different approaches is one way to obtain a system that handles a wider range of anomaly detection problems. The approach for classifier fusion in this experiment is to combine different kinds of classifiers. Another approach is to use a number of identical classifiers built on different datasets with different thresholds. This ensemble approach is commonly used for solving data mining problems and there is a wide range of methods for fusing the results [118].

The results show that combining the output from two different anomaly detectors can increase the total detection performance. The downside is that

the number of false positives also increases. If another classifier fusion scheme, such as *and* or *average*, were to be used, the number of false positives could be reduced at the cost of fewer anomalies found. For example, using the *and* operator, would mean that only the anomalies found by both detectors would be reported to the operator.

5.2.5 Summary and Conclusions

In this experiment, previous work on detecting behavioural anomalies in video surveillance data was extended to a more complex domain. The main difference in the domain is the number of objects present in the scenes and the variance in the data. To handle the increased complexity, we propose that both the anomaly detection methods from [28] should be extended and used together to find as many anomalies as possible without increasing the number of false alarms. The SBAD method was extended with more state classes and temporal state transitions, which increased the detection performance, but there are still some aspects of the method that should be further investigated. These aspects include, for example, how to combine the internal classifiers, how to find appropriate thresholds, how to find the optimal ratio between the total number of states and the amount of training data and how to automatically generate the states in a state class. Furthermore, the momentary track feature method was expanded with a new GMM feature model capturing position and stationary time; this enables the method to detect objects that are stationary for an anomalous amount of time. There are, however, some aspects that should be subjected to a more thorough investigation. The threshold settings when fusing the three internal classifiers, the size of the grids used to reduce the complexity of the feature models, and how to use more than one observation when classifying a track.

The experiments show that combining multiple anomaly detection methods can increase the number of detected anomalies compared to using only one method. However, the performance is still a long way from that in the previous domain [28], where the number of false positives was fewer than 1% and all anomalies were detected. In this project, the best setup resulted in 17% false positives while detecting 70% of the anomalies. This amount of false positives makes the approach hard to use in real-world applications. There are several reasons for this; the quality of the dataset was much lower than in the previous project and the tracking subsystem was not able to maintain the tracking during the lifetime of the objects. This is an important aspect that should be dealt with by improving the video tracker and the track correlator, and by making the anomaly detection methods more resistant to broken tracks. Due to the the low quality of the dataset, the anomaly detection methods should also be evaluated on a fully labelled synthetic dataset. This would enable the analysis of what properties in the track behaviour impact the detection performance most for each of the anomaly detection methods. Another area of improvement is the

fusing scheme used in the fusion module. In this experiment, a naïve approach for fusing the output from the detectors was used. Other, more advanced fusing schemes should be evaluated to see whether it is possible to decrease the percentage of false positives.

Chapter 6
Precise State-Based Anomaly Detection

This chapter addresses research objectives two, three, four and five. The content of the chapter is largely based on the two publications Brax et al. [31][1] and Brax et al. [32] where the latter has been submitted for publication. In the Brax et al. [31] paper, two new methods of detecting anomalies over time were proposed and evaluated. The two methods were based on precise and imprecise probabilities. The result was that the performance gain using the imprecise method was not big enought to motivate the increased complexity. Therefore, only the method based on precise probabilities is presented in this chapter. The main contributions of the chapter are:

- A precise method for handling anomalies that develop over time. The method employs more user-friendly parameters compared to the previous sliding window method.
- A method for automatically setting the alarm level using a validation dataset.
- A detailed evaluation of the SBAD method with real-world anomalies, which shows that the method can find up to 70% of the anomalies with, in this context, an acceptable number of false alarms.
- An analysis of the detection delay performance, which shows that the State-Based Anomaly Detection method is superior to two other anomaly detection methods.
- A study of real-world anomalies for the evaluation of anomaly detection methods.

[1]This publication is based on a collaborative endeavour between the author and Alexander Karlsson, where both persons contributed equally. The material from this publication is presented in sections 6.2 to 6.3.

- A feasibility assessment of the use of the SBAD method in the maritime domain.

6.1 Introduction

The *State-Based Anomaly Detection* (SBAD) method described earlier uses a sliding window approach to 1) detect anomalies that evolve over time and 2) to suppress false alarms. However, the sliding window approach has a number of problems, first, it is not clear how the detection threshold and window size are related and, second, it is hard for an operator to know how to set these parameters. This chapter proposes an extension of the original SBAD method using the Bayesian combination operator (cf. [14, 13, 81]) to cope with this problem. The performance of the proposed method is compared to the sliding window approach, as well as to two other anomaly detection methods. The proposed method is also evaluated together with the sliding window approach on a very large real-world dataset, where the anomalies are constructed together with subject matter experts to be as representative of real-world anomalies as possible.

6.2 Precise Anomaly Detectors

This section elaborates on how the normal models from the SBAD method (see Section 4) can be used in order to construct precise anomaly detectors.

6.2.1 Precise Anomaly Detectors

Consider the case where a new observation $\mathbf{y}$ is received from a sensor observing an object, the corresponding evidence regarding whether the object is anomalous or not can be constructed in the following way. Let $p(\mathbf{y})$ be any of $p_{Occ}(\mathbf{y})$ (see Equation 4.1) or $p_{Tr}(\langle \mathbf{y}_t, \mathbf{y}_{t+1} \rangle)$ (see Equation 5.1). Intuitively, the less probable it is to observe a specific $\mathbf{z}$, with respect to the normal model $p(\mathbf{z})$, the *stronger evidence* for anomaly. Let us introduce a random variable X for an object being anomalous or normal, i.e., a state space $\Omega_X = \{a, n\}$ (anomaly or normal). Based on this line of reasoning, evidences $\hat{p}(\mathbf{z} \mid X)$ can be constructed in the following way:

$$\hat{p}(\mathbf{z} \mid a) \triangleq \begin{cases} 0.5 + \frac{T - p(\mathbf{z})}{T} s_a & p(\mathbf{z}) < T \\ 0.5 - \frac{p(\mathbf{z}) - T}{\left(\max_{\mathbf{z} \in \Omega_{\mathbf{z}}} p(\mathbf{z})\right) - T} s_n & p(\mathbf{z}) \geq T \end{cases}, \tag{6.1}$$

$$\hat{p}(\mathbf{z} \mid n) \triangleq 1 - \hat{p}(\mathbf{z} \mid a), \tag{6.2}$$

where s_a and s_n are parameters that model the maximum possible strength that an evidence can constitute for anomaly respectively normality, and where

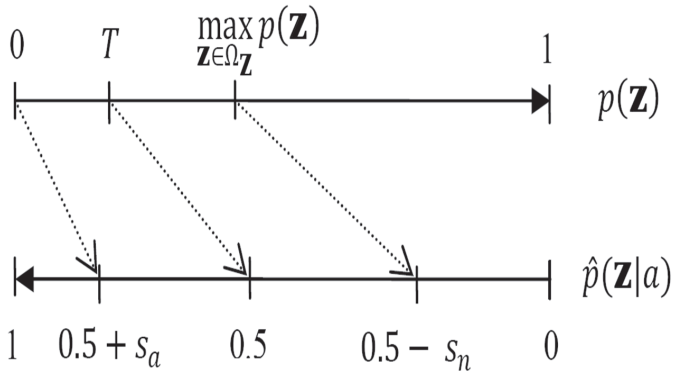


Figure 6.1: Mapping from $p(\mathbf{z})$ to $\hat{p}(\mathbf{z} \mid a)$.

T is a threshold that constitutes the limit between normality and anomaly. The mapping in Equation 6.1 is seen in Figure 6.1.

We see that when we have not made any observations at all in the normal model, we have the strongest possible evidence for anomaly and the other way around for normality. It can be somewhat counterintuitive to think of an observation $\mathbf{y}$ as constituting evidence for normality. However, if one wants an object to be able to "recover" from an anomalous state, such a modelling approach is necessary. Note that $s_n = 0$ implies there is no evidence for normality. The threshold T is a parameter that can be set with respect to the application at hand in order to adjust the sensitivity of the detector.

Now, at each time step t we obtain an observation $\mathbf{z}_t$, which we can use in Equation 6.1 to obtain a corresponding evidence $\hat{p}(X) \triangleq \hat{p}(\mathbf{z}_t \mid X)$. In order to simplify the notation, let us introduce:

$$\hat{p}_t(X) \triangleq \hat{p}(\mathbf{z}_t \mid X). \tag{6.3}$$

We can now use the $\Phi_{\mathcal{B}}$ operator in order to combine all evidences that have been obtained regarding a specific object into a joint evidence $\hat{p}_{0:t}(X)$:

$$\hat{p}_{0:t}(X) \triangleq \Phi_{\mathcal{B}}(...\Phi_{\mathcal{B}}(\hat{p}_0(X), \hat{p}_1(X))..., \hat{p}_t(X)), \tag{6.4}$$

where $\hat{p}_0(X)$denotes the *prior evidence* which is specified by the user.

6.2.2 Anomaly Classification

Let $\hat{p}(X)$ and $\hat{p}_{0:t}(X)$ denote evidence respectively joint evidence for any of the anomaly detectors that we have introduced. The question then is: When, in time, should a certain object be classified as an anomaly? In principle, such

a classification can be performed by introducing a threshold $T' \in [0, 1]$ and classifying the object as anomalous when:

$$\hat{p}_{0:t}(a) \geq T'. \tag{6.5}$$

Such a classification schema is not only dependent on the threshold T' but also on the maximum possible strengths s_a and s_n. Consider, for example, an object that has generated a large number of normal observations followed by only a few anomalous ones. In such a case, the joint evidence, before taking the anomalous observations into account, will constitute strong evidence for normality. Whether or not one classifies the object as an anomaly in such a situation depends on all parameters s_a, s_n and T'. For this reason, these parameters should be set jointly, by determining the number of *extreme anomaly evidences*, i.e., $\hat{p}(a) = 0.5 + s_a$, that is required, in order to classify the object as an anomaly, starting from the prior evidence $\hat{p}_0(a)$. Furthermore, in order to guarantee a certain level of reactivity of the detector, it is necessary to limit the joint evidence $\hat{p}_{0:t}(n)$ for normality to a minimum that is equal to the prior evidence:

$$\min_{\langle \mathbf{z}_1, \ldots, \mathbf{z}_t \rangle} \hat{p}_{0:t}(n) \geq \hat{p}_0(n), \tag{6.6}$$

for all tracks $\langle \mathbf{z}_1, ..., \mathbf{z}_t \rangle$. Such a limitation ensures a certain minimum degree of reactivity of the anomaly detectors. An example of the procedure is seen in Figure 6.2. From the figure, we see that if we want an object to be classified as an anomaly after eight extreme observations, we need to set $s_a \approx 0.2$. When s_a has been set by using our proposed method, one also needs to consider the maximum strength of an evidence for normality, i.e., s_n. The question that needs to be addressed when choosing such a strength, is to what extent should a normal observation influence the joint evidence towards normality. This can be modelled by considering the ratio between the strengths s_a and s_n.

The output from the precise occurrence and the precise transition detectors must be fused together before an alarm can be raised. In the sliding window approach, a weighted sum was used. This might, however, not be the best way of fusing detectors, because a detector that has a low weight might "pull down" the other detectors and, hence, decrease the chance of finding an anomaly (but also the risk of false alarms). If the detectors' output were overlapping, i.e., the detectors finding the same type of anomalies, the weighted sum method could be used, otherwise, if the detectors finds different types of anomalies, it should be enough that only one detector raises the alarm and, hence, the max operator should be used. The max operator outputs the same degree of anomaly as the detector with the maximum output. The precise occurrence and transition detectors have a partial overlap in their corresponding detection spaces. This is when an object travels outside the populated state-space in the normal model. This means that if an observation is received, which has no

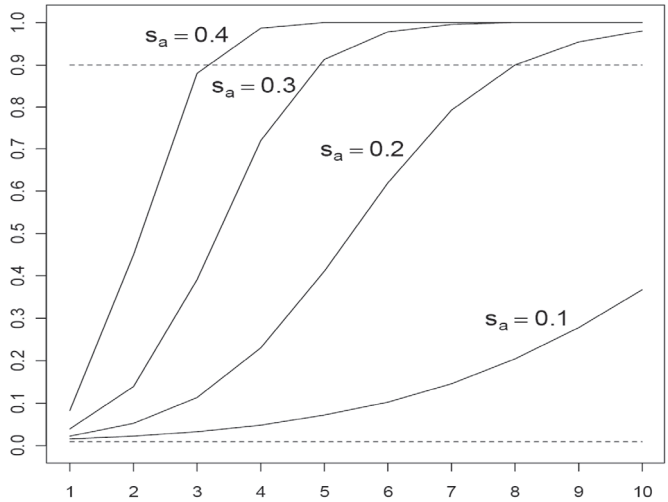


Figure 6.2: The figure depicts different choices of the parameter s_a where $T' = 0.9$ (upper dashed line). The x-axis shows the number of combinations and the y-axis shows the strength of the joint evidence $\hat{p}_{0:t}(X)$. The lower dashed line shows the prior evidence $\hat{p}_0(n)$.

corresponding occurrence or transition in the normal model, both detectors will raise the alarm. Since the overlap only exists under this premise, the max operator can be used.

6.3 Empirical Evaluation with Synthetic Anomalies

To evaluate the performance and feasibility of the precise detectors, a number of experiments were designed and carried out using real-world maritime AIS [54] data, where deliberate anomalies were introduced. In this section, the anomalies used in the evaluation have not been assessed by subject matter experts. The aim of these experiments is to compare the proposed approach with the previous one, as well as with other anomaly detection methods.

6.3.1 Datasets

Laxhammar et al. [88] introduced a novel approach for comparing the performance of anomaly detection methods, which we use in the first experiment. We use the same original dataset, as described by Laxhammar et al., for the two experiments. The dataset includes three weeks of AIS data recorded along the Swedish west coast and was preprocessed according to Section 3.3 by Laxhammar et al. The pre-processing resulted in a new dataset containing 2888

tracks with a total of 216717 observations. The new dataset was divided into one training set (2310 tracks) and one evaluation set (578 tracks). Each track contains a number of observations with attributes such as: time-stamp, latitude position, longitude position, speed and heading. To decrease the size of the original dataset, the observations were sampled with a distance of 200 m. The tracks were discretised into composite states, $\mathbf{Y}$, consisting of atomic states: position, velocity and heading. We discretise by using a positional grid with 45 by 45 cells around the area of interest (each cell is 304 by 198 meters), four velocity classes with limits 3, 15 and 25 knots and eight equally large classes for the course. For more information regarding the discretisation, see Brax and Niklasson [26].

6.3.2 Experiment 1 – Detection Delay

The aim of this experiment is to evaluate the reactivity of the anomaly detectors, i.e., *detection delay*. The experiment is equivalent to the one performed by Laxhammar et al. [88], where the authors compare two anomaly detection methods based on *Gaussian Mixture Model* (GMM) and *Kernel Density Estimator* (KDE). Assume that, for each track, anomalous observations are introduced after a random time point t to the end of the track by using a random walk function. Since each modified track produces a constant flow of anomaly observations from time point t to the end, the vessel can be classified as anomalous from time point t. The question now is: How long will it take for our set of anomaly detectors to classify the vessel as anomalous from the given time point? Clearly, the more reactive the anomaly detectors are constructed, the shorter the delay for detecting the anomaly. However, by constructing highly reactive detectors, the likelihood of classifying a normal vessel as anomalous before the vessel has started producing anomalous observations increases, resulting in a false alarm. As mentioned before, the reactiveness of our anomaly detectors can be modelled by the parameters s_a, s_n and T'. In the experiments conducted by Laxhammar et al. [88], the threshold values for the GMM and KDE methods were set such that 1% of the normal trajectory segments contained one or more anomalous observations prior to time point t. This can also be interpreted as the classification of 1% of the evaluated tracks resulted in a false alarm, i.e., a normal track classified as anomalous. To compare our proposed methods with the those evaluated by Laxhammar et al., we set the parameters s_a, s_n and T' using a linear search over the parameter space, to generate at most, 1% false alarms in the evaluation dataset. We also set the parameters (see Section 5.1.2.3) for the sliding window approach to generate at most 1% false alarms. The test dataset included 10000 tracks that were constructed as follows: randomly select one track from the evaluation dataset[2], then, intro-

[2]The same track can be selected multiple times.

Table 6.1: Parameter settings for precise and sliding window detectors. The parameter n_{ext} corresponds to the number of extreme observations required to reach the threshold T' starting from the prior evidence $\hat{p}_0(a)$. n_{ws} corresponds to the sliding window size. The parameters for the sliding window approach are described in Section 5.1.2.3.

Detector	n_{ext}	T	$\hat{p}_0(a)$
Precise	2	$\min_{\mathbf{y} \in \Omega_{\mathbf{y}}} p(\mathbf{y})$	0.01

Detector	n_{ws}	T	T_{al}
Sliding Window	5	$\min_{\mathbf{y} \in \Omega_{\mathbf{y}}} p(\mathbf{y})$	3

Table 6.2: Results of detection delay experiment. The table shows the mean and median number of observations after the time point t that the detectors need in order to classify the track as anomalous. For the mean values a 95% confidence interval is also presented.

Detector	Mean	Median	False alarms
Sliding window	4.19 $\pm$0.05	3	0.94%
Precise	3.60$\pm$0.05	2	0.93%

duce an anomalous segment as an random walk from time point t as described above.

6.3.2.1 Results

The parameters that performed best for each method are shown in Table 6.1. The detection delay results for the methods using the parameters in Table 6.1 are shown in Table 6.2.

According to previous experiments by Laxhammar et al. [88], the GMM and KDE methods needed 17.72 and 17.43 observations, respectively, before classifying a track as anomalous. Both methods had a median of 12 observations. Our proposed precise approach outperforms the previous methods based on GMM and KDE and is slightly better than the sliding window approach. A possible explanation of why the results of our proposed methods are better than the GMM and KDE is that our methods have the ability to capture the behaviour over time, i.e., anomalies occurring in multiple time steps can be accumulated.

6.3.3 Experiment 2 – Precision and Recall

The aim of this experiment is to evaluate the precise anomaly detectors' ability to detect anomalies in the form of a sequence of anomalous observations. To evaluate the performance, two commonly used measures from *data mining*, namely, *precision* and *recall* [132], are employed. Assume that the true class of each track in the test dataset is known. This information can be used to evaluate the performance of our detectors by using precision and recall:

$$Precision \triangleq \frac{n_a^t}{n_a^t + n_a^f}, \tag{6.7}$$

$$Recall \triangleq \frac{n_a^t}{n_a^t + n_n^f}, \tag{6.8}$$

where n_a^t denotes the number of tracks that have been correctly classified as anomalous, n_a^f denotes the number of tracks that have been misclassified as anomalous and n_n^f is the number of tracks that have been misclassified as normal. Note that an optimal detector has a precision and recall of one.

The parameters for this experiment can be found in Table 6.3. Due to the possibility of anomalies in the training data, we want to be able to adjust the sensitivity of the detectors in such a way that anomalies in the test data can be detected. In essence, a normal model $p(\mathbf{z})$ induces a partial ordering among the observations $\mathbf{y}$ with respect to normality. Such an ordering can be used to determine the sensitivity of the detectors by setting the threshold T (see Equation 6.1) to $p(\mathbf{z})$ where $\mathbf{z}$ belongs to a specific level T_{lev} in the ordering. Consider the following case:

$$\left\{\mathbf{z}_1^0, ..., \mathbf{z}_m^0\right\} \prec \left\{\mathbf{z}_1^1, ..., \mathbf{z}_{m'}^1\right\} \prec \left\{\mathbf{z}_1^2, ..., \mathbf{z}_{m''}^2\right\} \prec ..., \tag{6.9}$$

where:

$$\mathbf{z}_\bullet^i \prec \mathbf{z}_\bullet^j \text{ iff } p\left(\mathbf{z}_\bullet^i\right) < p\left(\mathbf{z}_\bullet^j\right). \tag{6.10}$$

If we set $T_{lev} = 2$, resulting in $T = p\left(\mathbf{z}_\bullet^2\right)$, then observations $\mathbf{z}_\bullet^0$ and $\mathbf{z}_\bullet^1$ yields anomaly evidences (see the mapping in Figure 6.1). $\mathbf{z}_\bullet^i$ denotes all the observations in the $i : th$ probability group. Let us define three sets of parameter settings, denoted by A, B and C, where T_{lev} is varied for the precise detector.

T_{lev} is also varied for the sliding window detector. Let us assume that ten consecutive observations that we have not seen before are regarded to be an anomaly. The parameter n_{ext} corresponds to the number of extreme observations required to reach the threshold T' starting from the prior evidence $\hat{p}_0(a)$. Hence, we set the parameter $n_{ext} = 10$ for the precise detectors. Similarly, for the sliding window detector, we set the detection threshold $T_{al} = 10$. The window size parameter n_{ws} is set to 20. This is mainly due to the results of the first experiments where the optimal window size was approximately $2 * T_{al}$. The

Table 6.3: Parameter settings for the Precise and Sliding Window detectors.

Detector	n_{ext}	A	B	C	$\hat{p}_0(a)$
Precise	10	$T_{lev} = 1$	$T_{lev} = 5$	$T_{lev} = 10$	0.01

Detector	n_{ws}	A	B	C	T_{al}
Sliding Window	20	$T_{lev} = 1$	$T_{lev} = 5$	$T_{lev} = 10$	10

lower evidence bound, $\hat{p}_0(n)$ in Equation 6.6 is set to 0.01 for all detectors. If the thresholds is set too high, for example, $e_{ext} = 20$, the detectors would have problems detecting the anomalies. The opposite, a too low threshold, would result in the detector finding the anomalies but also classifing a number of normal tracks as anomalies.

Now, in order to evaluate the design choice of the detectors with respect to precision and recall, a new test dataset needs to be constructed, where some tracks are anomalous. Since we have implicitly defined what we regard as an anomaly, i.e., ten consecutive observations not previously seen, let us introduce anomalies by skewing ten consecutive observations from some random start position. We perform the skewing of each point according to Figure 6.3. The maximum Δ is set to 500 m. The main difference between the datasets in exerpiment on and two is that, in the second dataset, the vessels that start to behave anomalously, will behave normally again after 10 observations.

We use the same training dataset as in the first experiment and sample the evaluation dataset to construct the test dataset. The test dataset has 200 tracks labelled normal and 200 tracks labelled anomalous, i.e., tracks where we introduce anomalies as described above.

6.3.3.1 Results

The results are presented in Table 6.4. We see that the sliding window approach performs best with respect to precision (0.95). However, the recall for the corresponding setting A is low compared to settings B and C. In fact, parameter setting A is not beneficial for any of the methods, due to the low recall. We also see that as T_{lev} increases, i.e., T increases in Equation 6.1, the recall increases due to the fact that we obtain more anomalous evidence.

After running the experiments with the real-world anomalies (see Table A.2), some additional conclusions about the sliding window approach were drawn, namely that the max operator should be used instead of the weighted sum because of an large increase in number of true positives compared to the small increase in false positives. Another conclusion is that that the same state detector should not be used in this domain due to a very high number of false

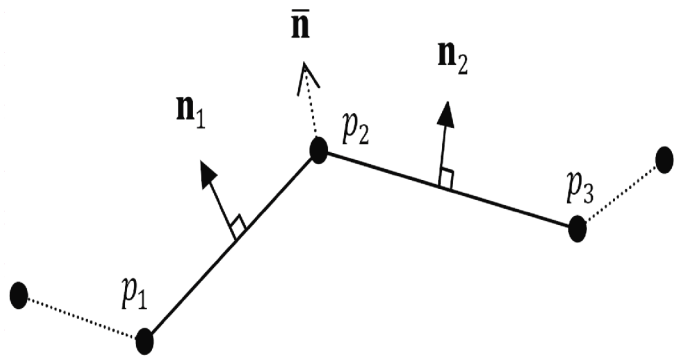


Figure 6.3: The point p_2 becomes skewed to a new point $p_2' = t\overline{\mathbf{n}}$ where $\overline{\mathbf{n}} = (\mathbf{n}_1 + \mathbf{n}_2)/2$ and $t \sim Un[-\Delta, \Delta]$ ($Un(\cdot)$ denotes the uniform distribution).

Table 6.4: Results of the second experiment for the different parameters settings A, B and C.

Detector	Precision			Recall		
	A	*B*	*C*	*A*	*B*	*C*
Precise	0.90	0.87	0.84	0.26	0.98	0.98
Sliding Window	0.95	0.89	0.83	0.52	0.88	0.94
Sliding Window version 2	0.97	0.84	0.80	0.97	0.99	0.99

positives. Based on these conclusions, the sliding window experiments were run again. The differences between the old and the new experiments were that the sliding window approach now uses the max operator with the occurrence and transition detector and a window size of 12 instead of 20. With the new parameter configuration, the sliding window approach performed very well. A precision and recall at 0.97 (setting A) can be considered to be a very good result. One possible reason for the performance boost between the original sliding window approach and the second version is that the weighted sum fusion scheme used in the original version suppresses anomalies and, hence, fewer anomalies are found. Another reason is that the smaller window size is enough to find the anomalies, but not high enough to result in false alarms. For settings B and C the new version of the sliding window detector has a very high recall but suffers from higher false alarm rates, i.e. a lower precision compared to the precise and the old sliding window approaches.

6.4 Experiments with Real-World Anomalies

The maritime anomalies used in the experiments presented in Section 6.3 were synthetic and might represent anomalies that do not occurr in the real-world. Although they are very effective for evaluation, based on the results, it is hard to say anything about the method's applicability for detecting real-world maritime anomalies. In order to evaluate the State-Based Anomaly Detection (SBAD) ability to detect real-world anomalies, a new dataset was developed together with maritime subject matter experts (SME).

The concept of *anomaly* can sometimes be hard to grasp. For a SME, an anomaly might indicate something out of the ordinary while, in Chapter 2 the concept is defined as: *something that does not conform with the model of normalcy*. Anomaly detection approaches can be used to detect new, previously unknown behaviours. The problem is that by definition, it is impossible to create an evaluation dataset containing truly unknown vessel behaviours. On the other hand, the unknown behaviours could be variants of known behaviours with some unknown parameters. For example, a pilot boarding a larger vessel is often likely to occur in the vicinity of ports, but is very seldom likely to occur out on the open sea.

Anomaly detection approaches can also be used to find known behaviours that are hard or impossible to define in advance. In the example above, it might be hard to define all the areas where the behaviour of a pilot boarding a vessel is normal. When developing a real-world dataset, the premise is that the anomalies are behaviours that are known to the subject matter experts, but the exact parameters of the behaviours are unknown or hard to define.

6.4.1 The Process of Developing the Real-World Maritime Dataset

The basis of the new dataset is 50 days of AIS data recorded along the Swedish west coast. The AIS data should reflect the normal behaviour of vessels, but might also include some anomalies. The AIS data is unlabelled, i.e., there is no label indicating whether an AIS report is normal or anomalous. This should not be a problem, because the SBAD method can handle training datasets with anomalies as long as the number of normal instances is dominant. The original AIS dataset is preprocessed and the resulting dataset is used to create various training and evaluation datasets with normal instances. To create an evaluation dataset with anomalies, a number of classes of potentially interesting anomalies are collected from the scientific papers and notes of two maritime security workshops. These anomaly classes are then refined and made more concrete, before being presented to maritime SMEs, who evaluate the relevance for each anomaly class. The final step is to create a number of concrete instances of each relevant anomaly class. These instances will form the part of the evaluation dataset that contains anomalies. Note that the meaning of "real-world

anomalies" in this work is not equivalent to *recorded data with manually labelled anomalous tracks*. Instead, the meaning is vessels that behave in ways that are, by subject matter experts, considered anomalous and interesting to detect. The dataset is still synthetic, but based on interesting behaviours from a SME's point of view; the anomalies in the dataset can be considered to be as close to real-world anomalies as possible, given the nature of the problem.

6.4.2 Maritime Anomalies

The available datasets are a limiting factor, when deciding which anomaly classes to use. The AIS data mainly includes kinematic data, type of ship, status information, voyage information and vessel dimensions. Therefore, the possible anomaly classes are limited to anomalies that include one or more of these attributes. It is not meaningful to create anomaly classes for anomalies that cannot be detected with the available datasets. For example, anomalies related to the crew of the vessels require crew rosters which were unavailable for this study.

The first step is a literature review, to identify anomaly class candidates. The two main sources of information found in the reivew are reports on two workshops held in Sweden [134, 12] and Canada [115]. These workshops included SMEs from various authorities and agencies and resulted in lists of maritime anomalies interesting for the SMEs. The Swedish workshop was part of a pre-study related to the Swedish Civil Contingencies Agency (SCCA) and the Department of Homeland Security (DHS) security call 2008 [39]. The aim of the workshop was to identify user requirements for a technical solution that could increase maritime domain awareness in Sweden and the United States. The user requirements included a number of potential *anomalies*, also called *early warnings*, which were relevant to the involved parties. The attendees of the workshop were subject matter experts and stakeholders from various Swedish authorities with different responsibilities in the maritime domain. During the workshop, 75 anomalies were identified, categorised and prioritised. Figure 6.4 presents an overview of the top 31 anomalies.

The identified anomalies were categorised as follows:

Tampering: Anomalies based on intentional concealment of current activity.

Owner/crew: Anomalies based on characteristics regarding people related to the vessel.

History: Anomalies based on data about the vessel, recorded in different databases.

Rendezvous: Anomalies based on vessels' encounters with other vessels or objects.

Movement: Anomalies based on the kinematic behaviour of vessels.

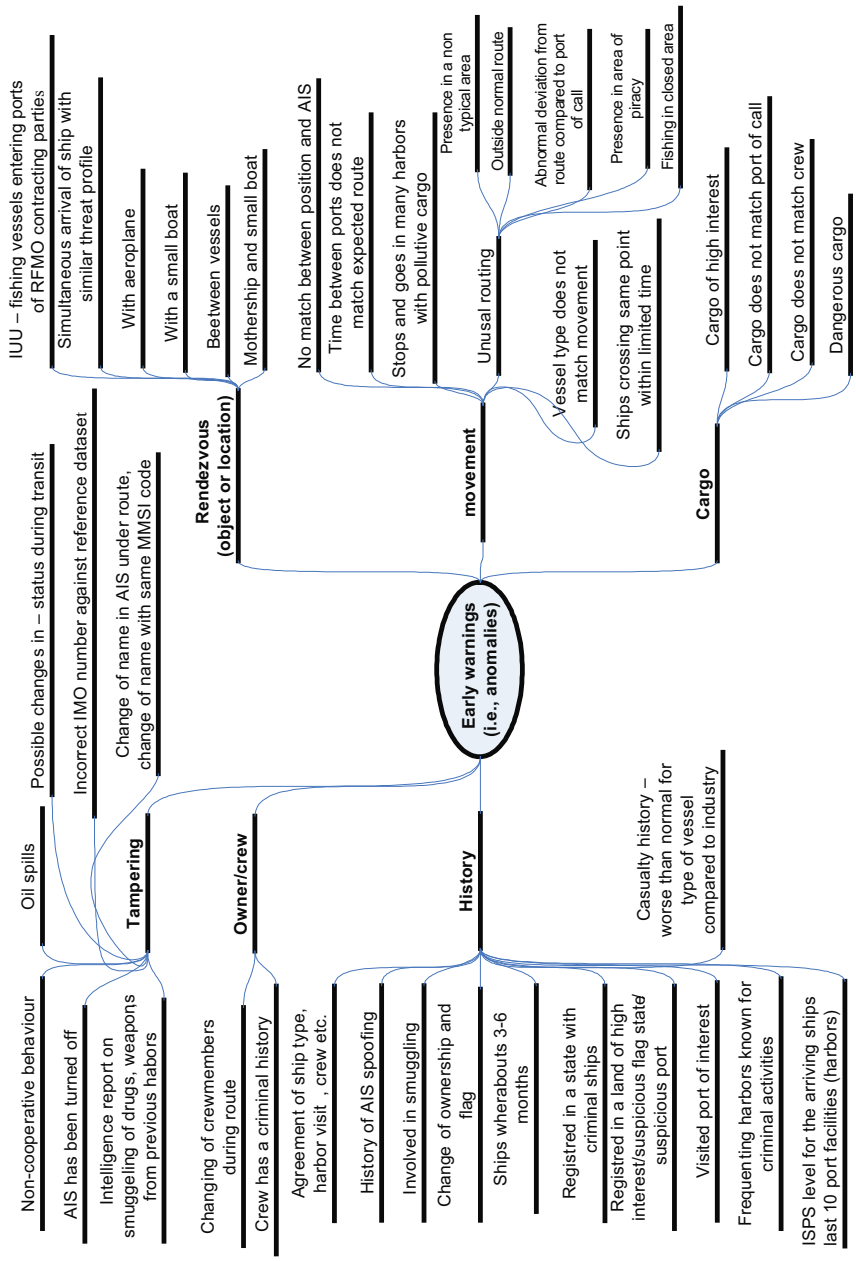


Figure 6.4: Overview of the identified anomalies in the Swedish workshop [134]. Figure adapted from [134].

Cargo: Anomalies based on the type of cargo the vessels are transporting.

The movement category is the most relevant category one for this study. The anomalies identified in this category are based on the attributes in the available AIS dataset. Some of the rendezvous anomalies could also be interesting but multiple object anomalies are not considered in this experiment.

The Canadian workshop [115] was part of the work on developing a Collaborative Exploitation Framework (CKEF). The aim of the CKEF was to support the analyst's knowledge management, to achieve maritime domain awareness. The workshop was conducted together with Canadian maritime subject matter experts for the purpose of capturing and documenting their know-how and collecting requirements for a rule-based expert system. One result of the workshop was a list of kinematic and non-kinematic anomalies. The kinematic anomalies were related to the course, speed, manoeuver, location and reporting of the vessel of interest, while the non-kinematic anomalies were related to passengers, crew list, ship signature, cargo, as well as next and last port of call. Based on the results of the workshop, the anomalies were refined and categorised, as shown in Figure 6.5 [116]. For the maritime anomalies dataset the anomalies in the Behaviour category in Figure 6.5 are the most relevant for this study.

Based on the anomalies identified in the Swedish and Canadian workshops, a number of potentially interesting anomaly classes were defined. These anomalies were then presented to two maritime domain SMEs, to obtain an assessment of the usefulness of being able to detect each of the anomaly classes, as well as more details about possible anomaly instances. Table 6.5 lists the ten defined anomaly classes, as well as their corresponding anomalies from the workshops. The rationale behind the anomaly classes is that they should be able to be detected, given the AIS dataset, and they should be suitable for data-driven anomaly detection. For example, it is more suitable anomalies such as *Change of Flag* or *Visit port of interest* with knownledge-driven (e.g. rule-based) detection approaches.

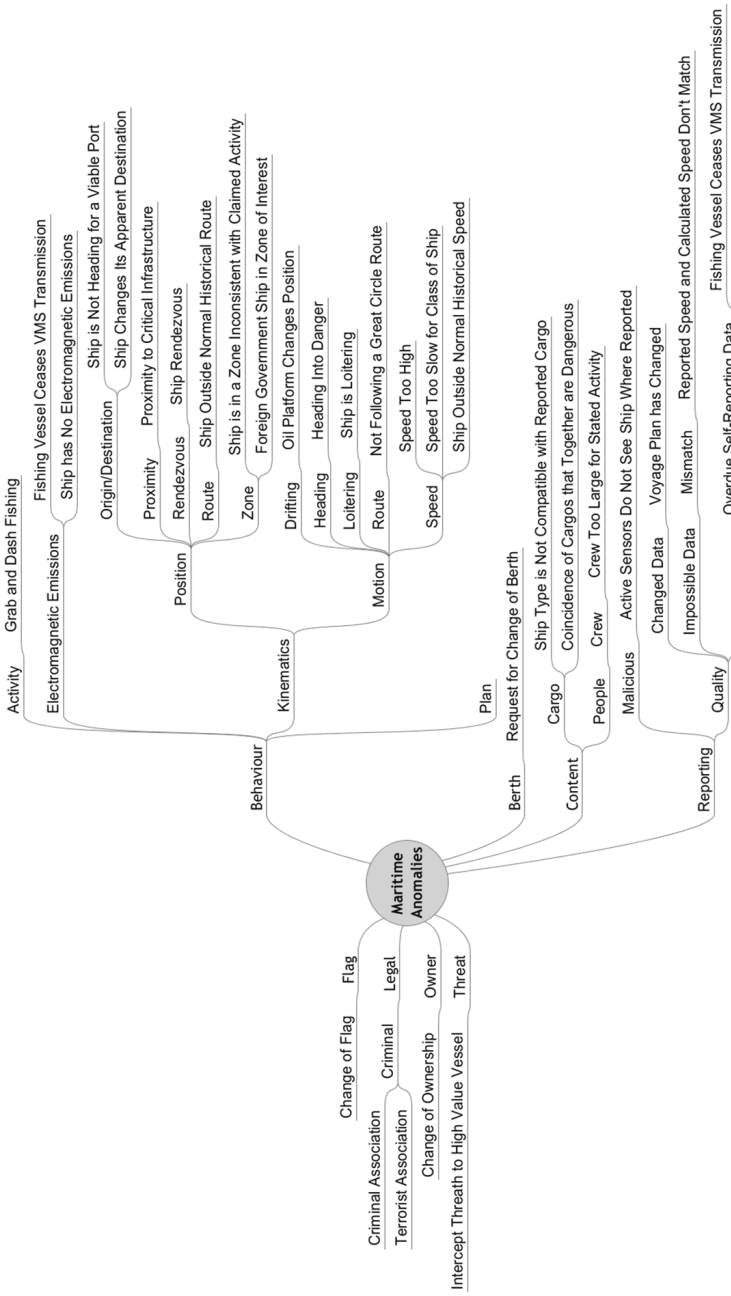


Figure 6.5: Categorisation of maritime anomalies based on the maritime situational taxonomy by Roy and Davenport [116]. Figure adapted from [116].

Table 6.5: Suggested anomaly classes and references to the Swedish and Canadian workshop.

No	Anomaly Class	Swedish workshop reference	Canadian workshop reference
1	Unexpected stop	S1	C1
2	Passenger vessel on a new route	S2, S3	C2
3	Large vessels in unusual locations	S2, S4	C2
4	A vessel of type X behaves like a vessel of type Y	S5	
5	A vessel misses a turn	S1, S2	C2
6	A vessel makes a strange turn in an odd location	S1	
7	Passenger vessel at strange time of day	S3	C2
8	Vessel travels too slow/fast	S1, S5	C1, C3, C4
9	Strange manoeuvre behaviour	S1, S5	
10	Vessel travels in circles and end up on land	S1, S2, S3, S5	

Table 6.6: Mapping of anomalies from the Swedish workshop.

ID	Swedish workshop anomaly
S1	No. 51 - Unusual speed, direction or turn
S2	No. 8 - Unusual routes
S3	No. 17 - A specific vessel deviates from its usual path
S4	No. 57 - Those with known dangerous cargo (potential to harm ports)
S5	No. 15 - Dynamics for the vessel do not match the characteristic manoeuvrability for that vessel type

Table 6.7: Mapping of anomalies from the Canadian workshop.

ID	Canadian workshop anomaly
C1	Speed Too Low for Type of Ship
C2	Ship Outside Normal Historical Route
C3	Ship Outside Normal Historical Speed
C4	Speed Too High

Maritime Anomaly Classes

This section presents a detailed description of the ten proposed anomaly classes from Table 6.5. The list presents some, but not all, the possible reasons and variants of each anomaly. This information was presented to the SMEs to help them assess which reasons and variants can be of interest.

1. Unexpected stop, a vessel stops unexpectedly.
   Possible reasons for stop: collision between vessels, grounding, engine problems, anchoring.
   Possible locations: shipping lane, in the port, along the coast, in open waters.

2. Passenger vessel on a new route.
   A passenger vessel deviates from its normal route.
   Variants: travelling on another side of an island, travelling to unusual destination.

3. Large vessels in unusual locations. Large vessels enter an area which they do not usually traffics.
   Possible locations: fishing areas, small ports, close to islands, nature protection areas.

4. A vessel of type X behaves like a vessel of type Y.
   Possible reasons: misconfigurated AIS transponder, deception for the purpose of misleading authorities while, for example, poaching.
   Variants: fishing vessel behaves like a passenger vessel, passenger vessels behave like a pilot, tanker behaves like a fishing vessel, tanker behaves like a transport vessel, etc.

5. A vessel misses a turn.
   A vessel follows the shipping lane towards a port but misses a turn where other vessels usually turn, the vessel travels towards a collision with land.

6. A vessel makes a strange turn in an odd location.
   A vessel travels straight towards land and makes an abrupt turn to avoid a land collision.

7. Passenger vessel at strange time of day.
   A passenger vessel travels its usual route, but at the wrong time of day.

8. Vessel travels too slow or too fast.
   A vessel travels too slow or too fast with respect to the normal speed for a certain type of vessel in a certain area.
   Possible reasons: some kind of problem, such as, engine failure, leakage or illegal activity underway, hijacking and vessel in a hurry (might not be interesting for the surveillance system operator).

9. Strange manoeuvre behaviour.
   A vessel turns often and in a strange way.
   Possible reasons: Malfunction in the on-board navigation or steering system, navigation officer asleep or drunk. A misconfigurated AIS transponder.
   Variants: A cargo vessel behaving like a sailing vessel, etc.

10. Vessel travels in circles and end up on land.
    Possible reasons: Navigation officer asleep or drunk.

Maritime Subject Matter Experts

Two maritime SMEs from the Swedish Maritime Administration (Sjöfartsverket[3]) were interviewed in order to obtain information about the feasibility of the ten identified anomaly classes as well as other useful information about their work at the VTS (Vessel Traffic Services) Centre. The two SMEs are stationed at one of the administrations VTS Centres. The VTS Centres are responsible for managing the commercial maritime traffic along the Swedish coast. The focus of the service is to improve vessel traffic safety and efficiency and at the same time, protect the environment [4]. The first SME was the current manager of the VTS Centre; he also had many years of experience as a VTS operator and as a pilot captain. The second SME was in charge of research collaboration in various research projects sponsored by the European Union. In these projects, future VTS capabilities, such as electronic navigation (E-Nav) are to be developed. The second SME also had many years of experience as a VTS operator.

Interview with the SMEs

During the interview with the SMEs, the ten anomaly classes were presented one at a time and the two SMEs were asked if anomalies of each class could occur in the real world and, if so, would detecting such anomalies with an automatic system be interesting. The SMEs also had the opportunity to comment on the details of each anomaly class as well as suggesting new anomaly classes. Table 6.8 summarizes the answers from the SMEs. It can be argued that

[3]See www.sjofartsverket.se for more information about the Swedish Maritime Administration.

it could have been better not to present any pre-defined anomaly classes and let the SMEs suggest their own classes. However, earlier experiences with SMEs indicated that it can be very difficult for SMEs to draw up an extensive list of anomaly classes during an interview. Moreover, even if the SMEs are asked in advance to suggest a number of anomalies, it can be very hard and takes a lot of time. Therefore, a number of pre-defined classes were presented. The result of the interview was that eight out of ten anomalies could occur in the real world and were interesting for the SMEs. Therefore, the dataset used in the experiment includes all the anomaly classes listed in Table 6.8 except for numbers two and six.

Beside the questions about the anomaly classes, others about the classification of vessels were also put to the SMEs. This information can be used to set the discretisation parameters for some of the atomic state classes. The SMEs were asked to give numbers for the speed states *slow*, *medium* and *fast* for vessels as well as for the size states *small*, *medium* and *large*. The answers to this question can be found in Tables 6.9 and 6.10. The SMEs were also asked whether an automatic detection system, such as the one presented in this work, could benefit them when solving everyday tasks. The answer was that they do not usually use the automatic alerting functionality in the VTS software. This functionality was considered to be too limited to be of any use. More advanced alerting where the system can learn from data was, however, very interesting. They argued that in cases of long shifts and when focusing on a single vessel that needs lots of attention, a system that could help them keep track of other vessels that deviate from normalcy would be of great use.

6.4.3 The AIS Dataset

The dataset used for the experiments was recorded by an AIS base station in Gothenburg, Sweden. The coverage of the base station is dependent on a number of factors, e.g., weather conditions, atmospheric attenuation, other radio traffic and so forth, and cannot easily be calculated. To avoid problems with vessels travelling outside AIS coverage, a smaller area, well inside the base station's coverage, was selected. Figure 6.6 shows the area used in the experiments.

The dataset includes a total of 50 days of recorded data from the 2nd February 2010 to the 6th April 2010. The data was recorded as raw AIS NMEA (National Marine Electronics Association) sentences [74] with a time-stamp for each sentence. An AIS NMEA sentence corresponds to one AIS report sent by a vessel. The size of the raw data is about 3.9 GB and includes approximately 57.3 million AIS reports. The raw AIS format was developed with the aim of being as compact as possible, which is due to the fact that the data is sent over low speed radio connections. The raw AIS data must be decoded before it can be used. See Algorithm 6.1 for an example of a raw AIS report and its decoded counterpart.

Table 6.8: Results of presenting the ten anomaly classes to the two SMEs.

Anomaly Class No	Occurs in the domain?	Interesting to detect?	Comment
1	Yes	Yes	Not interesting in anchoring areas. Does not apply to tugs, pilots, police and Coast guard.
2	Yes	Not for the VTS Service	Ships usually use Böttöleden, if they use Torshamnsleden it is an anomaly.
3	Yes	Yes	Outside the shipping lane is an unusal area to travel for large vessels.
4	Yes	Not for the VTS Service but for the Coast guard.	This is related to security (Coast guard) and not safety (VTS Service)
5	Yes	Yes	Two examples of locations where this can occur were given.
6	Yes	No	Can be considered as a special case of anomaly class five.
7	Yes	Not for today's VTS Service but for future E-Nav services	Only interesting for transport vessels, not for passenger vessels.
8	Yes	Yes	Interesting everywhere but especially around Torshamnspiren (no speed limit in the area) and around bunker vessels.
9	Yes	Yes	Evasive behaviour interesting. Vessels too close to the Vinga lighthouse (10 km is minimum, otherwise risk of collision with oncoming traffic).
10	Yes	Yes	Occurs sometimes, an indication is that the vessel crosses the center line in the shipping lane.

Table 6.9: Assessment of suitable limits for the speed states from the SMEs.

State	Speed	Comment
Slow	< 10 knots	
Medium	10–17 knots	
Fast	> 17 knots	

Table 6.10: Assessment of suitable limits for the size states from the SMEs.

State	Size	Comment
Small	< 120 m	Feeding boat and smaller vessels
Medium	120 – 200 m	Large passenger vessels
Large	> 200 m	Tankers and cargo vessels

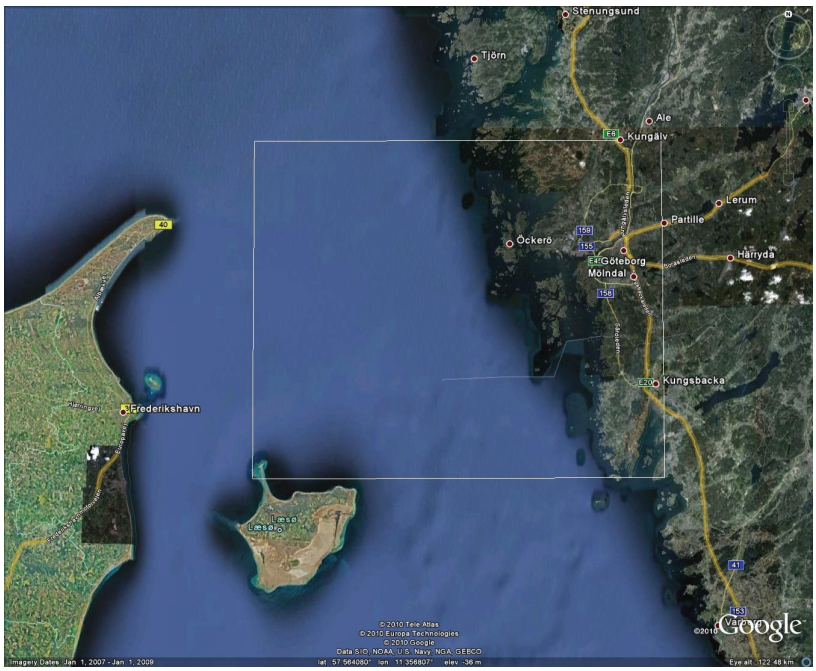


Figure 6.6: AIS base station coverage. Picture exported from Google Earth. ©2010 Tele Atlas, ©2010 Europa Technologies, ©2010 Google, Data SIO, NOAA, U.S. Navy, NGA, GEBCO.

Algorithm 6.1 Example of a raw and a decoded AIS report.

```
Time-stamp: 20100215 - 12:37:58.281
Raw AIS report: !ABVDM,1,1,,B,33u><;?PA30mK<
    BQ11n2B21j00u1,0*75

Decoded AIS report:
message type          = 3
repeat indicator      = 0
MMSI                  = 265522220
navagation status     = 15
rate of turn          = 129
speed over ground     = 6
position accuracy     = 0
longitude             = 11.670842
latitude              = 57.699773
course over ground    = 584
true heading          = 64
```

6.4.4 Data Pre-Processing

The raw AIS dataset must be preprocessed before it can be used in the experimental platform. The first step in the pre-processing is to decode all AIS reports. Each report includes a MMSI (Maritime Mobile Service Identity), which is the unique identifier for the vessel. Using the MMSI, all reports from a specific vessel can be grouped together. See Algorithm 6.2 for an overview of the pre-processing.

After the AIS reports are decoded, a number of filtering methods are applied to the dataset. First, all reports outside the area of interest are removed (see Table 6.11 for the corner coordinates of the area of interest).

The next step is to remove duplicate reports and resample at a larger time interval. For example, when a vessel is docked at the harbour it sometimes continues to broadcast AIS reports, which are duplicate reports with the same position and the speed set to zero. The update rate of AIS reports can be very high, which leads to very large datasets. To create a more manageable dataset, the dataset is resampled in such a way that there is a minimum distance between two subsequent reports. This means that some reports are removed. The resample function uses two parameters, the minimum distance between two subsequent reports, T_{dist} (set to 100 m in all experiments) and a speed threshold, T_{speed} (set to 0.01 knots in all experiments). The function first checks the distance between two reports. If the distance is below the threshold T_{dist} and the speed is above the threshold T_{speed}, the second report is removed and the next one is fetched from the list of unprocessed reports. If the distance is

Algorithm 6.2 Pseudo code for the pre-processing of the AIS dataset.

```
for each raw AIS data file
  for each AIS report
    Decode AIS report
    Add/store decoded information in a XML file for
       each MMSI
  end
end

for each XML file
  Load all reports in file into memory
  Remove all reports outside the area of interest
  Remove duplicate reports and resample
  Split info multiple tracks
  Save track in new XML file(s)
end
```

Table 6.11: Area of interest, used in pre-processing to remove AIS reports outside this area.

Minimum Latitude	57.34°
Maximuim Latitude	57.87°
Minimum Longitude	10.9°
Maximum Longitude	12.1°

above the threshold T_{dist}, the report is stored and used as a reference when processing future reports. If the speed is below the threshold T_{speed} the vessel is assumed to be docked and the reports are sampled with a time interval $T_{minTime}$ (set to 60 seconds in all experiments) instead of a distance interval.

The last pre-processing step is to split the track into sub-tracks. Since each AIS data file holds all the reports for a specific MMSI from the entire 50 days of data, the file includes a number of passes throughout the area of interest. A vessel might arrive and leave Gothenburg harbour several times each week. To be able to sort the tracks and make the analysis of the anomaly and normal dataset easier, it is desirable that each track file only includes one single pass through the area, for example, one track might correspond to a vessel arriving in the harbour and another track might correspond to a vessel leaving it. To split the track into subtracks, the time duration between two subsequent reports is used. If the time duration is greater than $T_{duration}$ (set to 30 minutes in all experiments) the track is split into two tracks.

6.4.5 Evaluation Dataset

To evaluate the proposed approaches, a number of datasets were developed. See Table 6.12 for a complete list of all datasets used in the experiments. Based on the interview with the SMEs, the most interesting vessels to monitor are cargo, passenger and tanker vessels. Other vessel types, such as police, military, pilot and pleasure craft often behave in a more random way, and it is hard to model their normal behaviour. The task of the Swedish Maritime Administration does not include these kinds of vessels. Therefore, a dataset including only cargo, passenger and tanker vessels was created. This dataset is called the CPT (Cargo, Passenger, Tanker) dataset and was used to create all other datasets used in the experiment. The first step was to split the CPT dataset into three smaller ones. This was done using the SRSWOR (Simple Random Sample Without Replacement) algorithm [136] to randomly split the CPT dataset into a training dataset (containing 80% of the tracks in the CPT dataset), a validation dataset (containing 10% of the tracks in the CPT dataset) and an evaluation dataset (containing the last 10% of the tracks in the CPT dataset). The validation dataset is used for parameter and threshold tuning. To create the normal test dataset, the evaluation dataset was randomly sampled for 604 tracks. The anomaly test dataset with the anomalies listed in Table 6.13 was created by randomly sampling 100 tracks from the evaluation dataset for each anomaly (except for the missed turn and circle and land anomalies) and then adding an anomalous part to each track.

The pseudo code for creating the *unexpected stop* anomalies can be found in Algorithm 6.3. The parameter X was set to 100, according to Table 6.13, and the Y parameter was varied, according to Table 6.15, to generate four different datasets. The reason for this is to evaluate the method's ability to detect anomalies of different lengths.

Algorithm 6.3 Pseudo code for creating unexpected stop anomalies.

```
Randomly select X tracks from the evaluation dataset
for each track
  Find a random observation in the track within the
     area of interest and outside normal anchoring areas
  Create Y number of duplicates of the observation with
      speed set to zero
  Update the time-stamp on all duplicates
  Update the time-stamp on all subsequent observations
end
```

The next anomaly, *large vessels in unusual places*, was generated according to Algorithm 6.4. Anomalies are generated by finding small vessels and adding 200 m to the length attribute, making them large vessels. It is assumed that

Table 6.12: Preprocessed dataset statistics.

Dataset	Number of tracks	Description
Complete preprocessed dataset	12373	Complete dataset including all vessel types. Used for creating all other datasets.
CPT dataset	8867	Subset of the complete dataset including only the vessel types Cargo, Passenger and Tanker.
Training dataset	7093	Used for building the normal model. 80% of the CPT dataset.
Validation	887	Used for threshold settings. 10% of the CPT dataset.
Evaluation	887	Used for creating the final test dataset with anomalies and normal tracks. 10% of the CPT dataset.
Anomaly Test dataset	604	Used to evaluate the methods. Includes anomalies added to tracks sampled from the evaluation dataset.
Normal Test dataset	604	Used to evaluate the methods. Tracks without anomalies sampled from the evaluation dataset.

Table 6.13: Anomalies used in the evaluation.

Anomaly class No	Name	Number of instances in dataset
1	Unexpected stops	100
3	Large vessels in unusual places	100
4	Wrong type	100
5	Missed turn	2
7	Strange time	100
8	High/low speed	100
9	Strange manoeuvres	100
10	Circle and land	2

the small vessels do not traffic the same areas as large ones. This assumption may or may not be true, depending on which tracks are randomly selected from the dataset. Since this can lead to anomalies that closely resemble large normal vessels, such anomalies can be very hard to find. However, it is a simple method for generating these kinds of anomalies.

Algorithm 6.4 Pseudo code for creating large vessel in unusual places anomalies.

```
Randomly select X tracks from the evaluation dataset
for each track
  if vessel size < 70 m
    Make vessel large by adding 200 m of length
  else
    Select another track
  end
end
```

The *wrong type* anomalies are generated in a similar fashion. Algorithm 6.5 depicts the method for creating this kind of anomaly.

Algorithm 6.5 Pseudo code for creating wrong type anomalies.

```
Randomly select X tracks from the evaluation dataset
for each track
  Get current vessel type
  Randomly select a vessel type that is not equal to
     the current vessel type
  Change vessel type to the new type
end
```

The *missed turn* anomaly is a special type of anomaly. Based on the interview with the SMEs, there are two areas in which vessels sometimes miss turning. There are only two instances of this anomaly, and both are handcrafted to match scenarios where vessels miss a turn in these two areas.

The *strange time* anomalies are generated according to Algorithm 6.6. An offset, randomly selected in the interval, I, $[8, 16]$ hours is added to all observations of each track. This offset can for example "move" tracks from the morning to the afternoon or from the afternoon to the night, which might be anomalous.

The next anomaly, *unusual speed*, is generated according to Algorithm 6.7. The speed change factor, Z, is $\{-90\%, 90\%\}$. This means that the speed is either increased by 90% or decreased by 90% for each time step from t_{random}

Algorithm 6.6 Pseudo code for creating strange time anomalies.

```
Randomly select X tracks from the evaluation dataset
for each track
  Randomly select time offset in interval I
  for each observation
    Add time offset to time-stamp
  end
end
```

to $t_{random} + Y$. If the speed of the randomly selected track is already low, a decrease in speed will most likely not affect the speed state. The reverse is also true, a fast vessel made faster. This class of anomalies is thus potentially hard to find.

Algorithm 6.7 Pseudo code for creating unusual speed anomalies.

```
Randomly select X tracks from the evaluation dataset
for each track
  Randomly select whether the speed should increase or
     decrease
  Find a random observation in the track
  Change speed by a factor Z of Y subsequent
     observations
end
```

The *strange manoeuvre behaviour* anomaly is implemented according to Algorithm 6.8. The skewing is done according to the algorithm presented in Section 6.3.3.

Algorithm 6.8 Pseudo code for creating strange manoeuvre behaviour anomalies.

```
Randomly select X tracks from the evaluation dataset
for each track
  Find a random observation in the track
  Skew observation by Z for Y subsequent observations
end
```

The last anomaly class, *circle and land*, aims to simulate a vessel whose rudder is malfunctioning or the captain has fallen asleep on the bridge. This anomaly class is based on a real incident where the vessel ran in circles before it

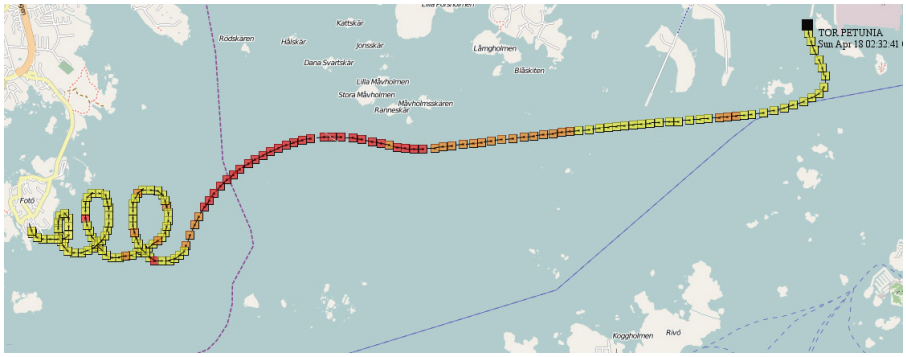


Figure 6.7: Example of the circle and land anomaly. The colour indicates the speed state of each observation. Map from www.openstreetmap.org, © OpenStreetMap contributors, CC-BY-SA.

collided with land. See Figure 6.7 for a graphical representation of the anomaly. Instances of this anomaly class are handcrafted, i.e., the position, speed, course and time-stamp are manually calculated for each observation. This is a very time consuming way of generating anomalies and, therefore, only two instances were created.

6.4.6 Experiment Process

All the experiments were performed using the same process. The only aspects that change are the parameters used when building the normal model and the threshold settings in the detectors. First, a normal model is built using the training dataset. The next step is to set appropriate thresholds in the detectors, which can be done in several ways. One is to use domain knowledge to set the thresholds in a way, that the chance of finding anomalies is maximised. The threshold settings will of course affect both the number of detected anomalies, as well as the number of *false alarms*. A false alarm is defined as *"a normal track classified as anomalous by the detector"*. This is the same as a *false positive*. The false alarm rate is calculated by dividing the number of false positives by the total number of normal tracks in the evaluation dataset. Setting the thresholds in such a way that the system finds all anomalies will result in a large number of false alarms. This is also true for the reverse. Setting the thresholds to minimize the number of false alarms will result in fewer detected anomalies. The false positives are called false alarms in this work, which is, however, a simplification, because whether the false alarm is truly a false alarm cannot be determined before it has been analysed by a human operator.

Another way of setting the thresholds is to use the *parameter search mechanism* in the experimental platform. By using this mechanism, it is easy to

evaluate a wide range of parameter settings. For example, all the combinations of a detection threshold between 1 and 250 and a window size between 1 and 250 can be evaluated, and the result sorted on the basis of the resulting performance. In the example, $250 * 250 = 62500$ combinations needs to be evaluated, which can be very time consuming. To reduce the search time, each parameter is increased by ten in the interval, i.e. $\{1, 11, 21, 31, ..., 250\}$, which decreases the number of resulting combinations to 625, a much more tractable figure. An evaluation typically takes one or two seconds and is implemented using a job queue where a number of working threads can do evaluations in parallel. Four different parameter combinations were evaluated in parallel on the computer[4] used for the experiments. The parameter search mechanism can be used in two ways. The parameter search can be used together with the evaluation dataset to find the parameter setup that results in the best performance. This works very well in theoretical evaluations and is used in many data mining experiments, but when dealing with a real-world systems the evaluation dataset is not available in advance. The only dataset available at system deployment time is the validation dataset that consists of data which has not been used when building the normal model. If this dataset is used with the parameter search mechanism it is possible to find threshold settings that result in a specific alarm rate. If the data in the validation dataset is representative of what can be expected in new data, the alarm rate in new data should be about the same.

6.4.7 Experimental Setup

After a number of initial experiments, the results showed some interesting aspects of the method. The first experiment was to build a normal model of all the vessels in the complete preprocessed dataset and then find the most anomalous ones. This was done by setting the T_{lev} threshold greater than zero. When a normal model was used without type information, the most anomalous vessels were the police, search and rescue, pilots, tows, pleasure craft and military vessels. Even some of the tankers and cargo vessels behaved anomalously when they entered open waters. This result was expected and demonstrates the SBAD's ability to filter out "strange" vessels among the ones considered normal. The next experiment was conducted with the CPT dataset. The training dataset was used to build the normal model and the complete evaluation dataset was used to evaluate the performance. With a 25 by 25 grid and a 10% alarm threshold, it was not possible to find more than 38% of the anomalies. After an analysis of the results, it was found that many of the vessels in the validation dataset were classified as anomalies and, hence, the detection thresholds were raised to avoid too many alarms. This, of course, made it harder to find the anomalies. There are several possible reasons for the alarms in the

[4]The evaluation system was built on an Intel Core i5 750, 2,66 GHz Quad-Core CPU with 8 Gb of RAM.

validation dataset. First, there might not be enough representative data in the training dataset, i.e., there are normal vessels that are not included in the training dataset. Another possible reason is that the normal model is too detailed, i.e., there are too many possible composite states compared to the amount of training data. This can be further investigated through experiments using fewer possible composite states, e.g., using a grid with fewer cells or disregarding one or several of the atomic states that are included in the composite states.

A number of different aspects of the SBAD method were evaluated in the experiments:

Temporal Detection Method: In all the experiments, both the sliding window method and the precise method were evaluated. Although, these approaches have also been evaluated in previous experiments, the results were based on a smaller dataset with only two types of anomalies.

Fusion Schemes: There are a number of ways of fusing the output from each anomaly classifier (Occurrence, Transition, SameState). In previous experiments with the sliding window approach, the classifiers were fused as a weighted sum; nevertheless, it is not clear how to set the weights besides through empirical studies. When the precise anomaly detectors were introduced, a theoretical analysis resulted in a fusion scheme based on the *max* function, i.e., if either of the classifiers signals an anomaly, an alert should be raised. This should also be true for the sliding window detector. To find out if that is the case, a number of fusion schemes, listed in Table 6.14, are evaluated. The SameState detector was not implemented in a precise version, because it adjusting the sensitivity of the detector in an intuitive way was not clear, compared to the occurrence and transition detectors. Two weighted sum approaches used in earlier experiments together with the max function approach and each classifier separately, are evaluated on the sliding window detector. For the precise detector, only the max function and each classifier separately, are evaluated.

Anomaly Length: This experiment aims at investigating how the length of the anomaly affects the performance. The different values for the anomaly length can be found in Table 6.15. This parameter can have a considerable impact on the results. Although, a long anomaly should be easier to detect, its occurrence in the real world might be more unrealistic compared to a shorter anomaly.

Grid Size: The number of columns and rows in the grid. With a finer grid, the size of the cells in meters is smaller than in a coarser grid. A finer grid can potentially capture more details in the training data, but requires more data to "fill" all the relevant cells. See Table 6.16 for a list of the different grid sizes evaluated in the experiments.

Composite States: In this experiment, a number of different composite state setups are evaluated to see how the inclusion of contextual information impacts the performance and the types of anomalies found. Table 6.17 presents the composite state setups evaluated in the experiments.

Highest Ranked Anomalies: This experiment aims to investigate how the false positives are distributed compared to the true positives. Each track classified as an anomaly is rated on the basis of how many observations ended up above the detection threshold. The list of rated tracks is then sorted in such a way that puts the most anomalous track at the top of the list. If the false positives end up at the top of the list, they might have a greater impact on the decisions based on the output from the detectors, compared to if they end up further down the list. A top list could be a good method for presenting anomaly alarms to an operator.

In all the experiments, the validation dataset was used to find threshold setups resulting in 1%, 5% and 10% alarms. Besides the grid size parameters used for the position state presented in Table 6.16, the following parameters were used for the discretisation in the atomic state classes:

Course: Discretised into eight atomic states with corresponding intervals: north (337.5, 22.5], north-east (22.5, 67.5], east (67.5, 112.5], south-east (112.5, 157.5], south (157.5, 202.5], south-west (202.5, 247.5], west (247.5, 292.5] and north-west (292.5, 337.5]. Each state is 45 degrees wide.

Speed: Discretised into four atomic states based on information from the SME interview. Table 6.9 shows the SMEs' assessment of the limits for the speed states: slow, medium and fast. The limit between stopped and slow was set to 0.01 knots to allow for some positional drifting.

Size: Also discretised into four atomic states based on the SMEs' assessments. See Table 6.10 for the limits between the states small, medium and large.

Time of day: Discretised into four atomic states with corresponding intervals, morning $(5, 11]$, noon $(11, 14]$, afternoon $(14, 18]$ and night $(18, 5]$.

Type: Based on the vessel type code in the AIS information, all of the vessels are one of the following types: unknown, reserved, other, wig, fishing, towing, sailing, pleasure, hsc, pilot, sar, tug, police, medical, passenger, cargo, tanker or undefined.

The performance measures used for all the experiments are precision, recall and false alarm rate. Precision and recall are defined in Equation 6.7 and in Equation 6.8.

Table 6.14: Fusion schemes for the sliding window and precise anomaly detectors.

Sliding Window Approach	
Fusion scheme	Formula for total Degree Of Anomaly (DOA)
Max: Transition, Occurrence	$\max(DOA_{Tr}, DOA_{Occ})$
Max: Transition, Occurrence, SameState	$\max(DOA_{Tr}, DOA_{Occ}, DOA_{SS})$
Only Transition	DOA_{Tr}
Only Occurrence	DOA_{Occ}
Only SameState	DOA_{SS}
Weighted sum: Occurrence, Transition, SameState	$\frac{2*DOA_{Occ}+DOA_{Tr}+DOA_{SS}}{4}$
Weighted sum: Occurrence, Transition	$\frac{DOA_{Occ}+DOA_{Tr}}{2}$

Precise Approach	
Fusion scheme	Formula for total Degree Of Anomaly (DOA)
Max: Transition, Occurrence	$\max(DOA_{Tr}, DOA_{Occ})$
Only Transition	DOA_{Tr}
Only Occurrence	DOA_{Occ}

Table 6.15: Parameters for anomaly length used for creating multiple datasets.

Anomaly length	Length in meters
25 observations	2500 m
50 observations	5000 m
100 observations	10000 m
200 observations	20000 m

Table 6.16: Number of grid cells and their corresponding size used in the grid size experiments.

Grid size (rows, columns)	Cell size (height, width)
25, 25	2.4 km, 2.8 km
12, 12	5 km, 6 km
6, 6	10 km, 12 km
3, 3	20 km, 24 km
1, 1	59 km, 71 km

Table 6.17: Composite state setups.

Name	Atomic states used in Composite state
All	Position, Course, Speed, Time Of Day, Size, Type
Only kinematic	Position, Course, Speed
No time	Position, Course, Speed, Size, Type
No type	Position, Course, Speed, Time Of Day, Size
No size	Position, Course, Speed, Time Of Day, Type
No position	Course, Speed, Time Of Day, Size, Type

6.5 Results

In this section, the results of all the experiments are presented together with comments on specific parameter settings. In parameter tables, such as Table A.1, the following notation is used: T_{al} denotes the alarm threshold for the sliding window approach, n_{ws} denotes the window size for the sliding window approach. n_{ext}^{Occ} and n_{ext}^{Tr} denote the number of extreme observations needed to reach the alarm threshold for the precise occurrence and precise transition detectors, respectively. In result tables, such as Table A.2, the abbreviations should be interpreted as follows: TP is true positives, TN is true negatives, FP is false positives and FN is false negatives. The *actual alarm rate* in the validation datasets refers to the number of alarms raised with the corresponding thresholds. The *false alarm rate* denotes the resulting false alarm rate given a specific threshold setup. This value of the actual alarm rate and the false alarm rate might differ between the validation and the evaluation dataset due to the random sampling of the source dataset and the specific threshold setup used in the experiment.

6.5.1 Computational Cost

The computational cost was measured on the experiments with a 6 by 6 grid using all the atomic states. The time required to build the normal model on the experiment computer was 82 seconds, during which 7093 tracks were pre-processed (discretised) and added to the normal model, by calculating both occurrence and transition statistics. In total, the tracks in the training dataset included 2.46 million observations, which means that the system was able to process approximately 30.000 observations per second. The time used for detecting anomalies in the 1208 tracks from the evaluation dataset was between 2 and 3 seconds.

6.5.2 Anomaly Detection Results

All parameter settings and results of the fusion scheme, anomaly length, grid size and composite state experiments can be found in Appendix A. In the fusion scheme experiments, only one alarm threshold for the validation dataset is reported in the results. The other thresholds were evaluated, but did not affect the relative results of the fusion methods. The anomaly length experiments were conducted using a 12 by 12 position grid setting. After the grid cell size experiments, it was concluded that a 3 by 3 grid yielded the best results. Therefore, all the composite state experiments were conducted with an anomaly length of 50 and a grid size of 3 by 3 grid cells. The results from using all the atomic states can be found in Table A.9, second part (grid size 3 by 3). In the experiments without positional information, the parameter search did not find any setups that resulted in actual false alarm rates higher than 2.59% and 3.27% for the sliding widows and precise approach, respectively.

6.5.3 Highest Ranked Anomalies

One way of presenting the result of the anomaly detection method to a human operator is as a top list with the most anomalous vessels. The operator can then choose to investigate the objects, based on how anomalous they are. By measuring how many observations are above the detection threshold for each vessel, it is possible to rank all vessels detected as anomalous. This list includes both true positives and false positives. It is important that the false positives do not dominate the top of the list, since, in which case, the operator might spend time investigating normal vessels. Figure 6.8 depicts a plot of the top 200 detections. The results are based on the experiments with the sliding window approach, a 5% threshold and a grid size of 3 by 3, as presented in Table A.9. In total, 437 vessels were classified as anomalous by the detector, 401 were true positives and 36 false positives. Figure 6.8 shows that there are 12 false positives present in the top 200 detected vessels. The remaining 24 false positives are distributed among the 237 remaining detections.

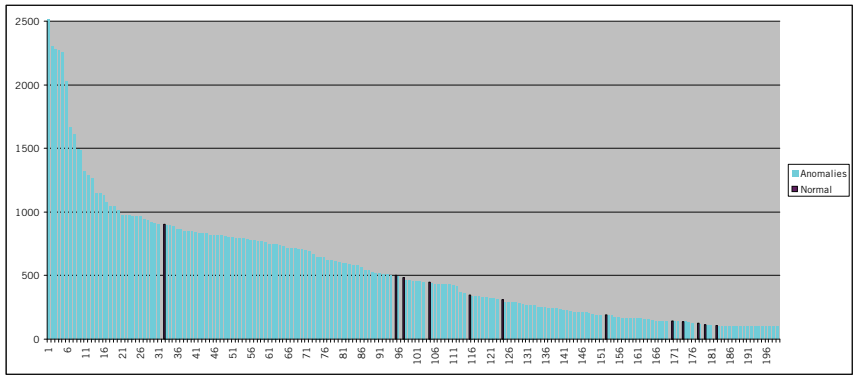


Figure 6.8: A plot of the 200 highest ranked vessels detected as anomalies.

6.6 Analysis of the Results

In this section the results of the experiments with real-world anomalies are analysed.

6.6.1 Computational Cost

The computational cost of the SBAD method is very low, at least compared to building GMMs with the EM algorithm. The computational cost of the SBAD method is low both when building the normal model and when detecting anomalies. The construction time for the normal model could be reduced even more; the loading and pre-processing of data are I/O bound, and with a faster I/O system, e.g., a Solid State Drive, the loading time could be decreased dramatically. The time required to generate transition and occurrence statistics could also be reduced, by implementing a multi-threaded approach instead of the current single-thread method. In theory, this could result in a 4x speedup, but in reality, based on the experience with multi-threaded evaluation, a 3x speedup should be possible on the four-core experimental system. Based on the experiments, it can be concluded that the SBAD method is most suitable for online detection of anomalies.

6.6.2 Fusion Scheme Experiments

Table A.2 indicates that the choice of fusion scheme greatly impacts the anomaly detection performance. As previously argued, the max operator is the most suitable operator when fusing the results of the detectors. The results show that the scheme which achieves the most detected anomalies is the max

operator with transition and occurrence detectors. This is true for both the sliding window and precise approaches. Another conclusion is that the same state detector does not work very well in this domain. It is the worst performer of the single detector schemes, i.e., achieves a false alarm rate of 99.67%. When the same state detector is used together with the max operator, the result is even worse. The weighted sum scheme with all three detectors performs quite well with respect to the false alarm rate, but the scheme does not result in as many found anomalies as the max operator with transition and occurrence detectors. The reason the weighted sum scheme to performs this well, despite the use of the same state detector, is due to the low weight on the result from the same state detector. Based on these outcomes, the same state detector should not be used at all in this domain and the max operator scheme with transition and occurrence detectors achieves the best performance with respect to the false alarm rate.

For the sliding window approach, when only the transition detector is used, the performance is the same as using the max operator with both the transition and occurrence detectors. This could be regarded as an unexpected result, but there are two explanations for this. First, the sensitivity of the detectors in this experiment is set to 0%, which means that everything not represented in the normal model is considered to be anomalous. It is also possible to individually adjust the sensitivity for the detectors in such a way that, for example, the 1% most uncommon observations in the normal model are considered to be anomalies. By setting the sensitivity to 0% for both detectors (to decrease the risk of false alarms), the occurrence detector will raise an alarm when a vessel ends up in a composite state that does not exist in the normal model. The transition detector will alarm for the same circumstance, but will also alarm when a vessel makes a transition between two composite states that do not exist in the normal model. The second explanation is that the two detectors use the same sliding window length. Therefore, the number of alarms from the transition detector inside the sliding window will always be greater or equal to the number of alarms from the occurrence detector. This means that in these experiments, it does not matter whether the max operator with the two detectors or only the transition detector is used.

For the precise approach, no single detector is as good as the two detectors combined with the max operator. This means that the two detectors finds the same anomalies most of the time, but in some cases only one of the detectors raises the alarm. Hence, they complement each other and it makes sense to use the max operator. This is a result of the second explanation described above, i.e., the two detectors use different settings for the number of extreme observations before giving an alarm and, therefore, the occurrence detector can raise the alarm without the transition detector also raising it.

The overall conclusion of this experiment is that the max operator with transition and occurrence detectors should be used for both the sliding window and precise approaches.

6.6.3 Anomaly Length Experiments

The anomaly length experiments demonstrate that the recall for the 10% alarm parameters increases when the length of the anomaly increases, which is also the case with the 5% alarm parameters. For the 1% alarm parameters, the recall stays roughly the same except for the anomaly length of 200, where the recall increases slightly. It is not surprising that the recall increases when the length of the anomaly increases. Since the detection parameters are the same for each alarm level, an anomaly with greater length will be easier to detect. The precision increases in the same way as recall, but not as much. Overall, the differences in the results between the various anomaly lengths are not very large. For example, consider the true positives for the 10% sliding window, which are 223, 323, 327 and 341 for the lengths of 25, 50, 100 and 200. There is a greater difference between the lengths of 25 and 50, but the difference between 50, 100 and 200 is not that big. This is quite expected, given the parameters found in the parameter search. For the sliding window approach with 10% alarms, the parameters found were an alarm threshold at 35 and a window size of 65. This means that the 25 anomalies would not trigger the detector, but all of the rest would. Given the results, any of the 50, 100 and 200 datasets would be a suitable candidate for further experiments. The anomalies in the 200 dataset would, of course, be the easiest to find. Another aspect of the choice of dataset for the rest of the experiments is the length of the distance, in meters, of each anomaly. Since the data is sampled with 100 meters between each observation and with an anomaly length of 25 observations, the actual length of the anomaly is 2.5 km. The other datasets result in anomalies of 5 km, 10 km and 20 km, respectively. Of these three, the 5 km anomaly is the one most likely to occur in the real world. Therefore, the dataset with anomalies of length 50 is chosen to be used in the remaining experiments.

6.6.4 Grid Size Experiments

The results of the grid size experiments show that the recall increases when the grid size decreases, except for the 10% alarm rate and the precise detection. However, the difference here is only two vessels, i.e., 422 detected anomalies with the 6 by 6 grid vs. 420 detected anomalies with the 3 by 3 grid. The recall for the 5% alarm level increases from about 0.45 to 0.65 (average of the two methods) when the grid size is reduced from 6 by 6 to 3 by 3. For the 1% alarm level, the recall always increases when the grid size decreases.

The precision follows the same pattern: for the 10% alarm threshold, the precision increases with smaller grid sizes, except from 6 by 6 to 3 by 3, where a small decrease in precision can be noted. For the 5% and 1% alarm thresholds, a minor decrease in precision between the 25 by 25 and the 12 by 12 grids can be noted, while the precision then increases with regard to the 6 by 6 and 3 by 3 grids.

In order to decide which grid size should be used in the remaining experiments, the number of wins, e.g., highest precision and recall for each alarm level is measured. The result is that, for the 6 by 6 grid, the 10% precise approach had the highest precision and recall (equals two wins) and the 10% sliding window approach had the highest precision (another win). This means that the 6 by 6 grid had a total of three wins. The 3 by 3 grid had wins for both methods in the 1% and 5% levels and for the sliding window approach in the 10% level. This equals 9 wins out of 12 and, therefore, the 3 by 3 grid will be used in all the remaining experiments. The 25 by 25 and 12 by 12 grid sizes had no wins.

The grid size experiments also show that the available training data is not sufficient to populate a normal model with a fine grid.

6.6.5 Composite State Experiments

The hypothesis in these experiments is that anomalies based on an object attribute that is not included in the normal model are harder to find than if the attribute is included. For example, if the size of a vessel is omitted from the composite state used in the normal model, anomalies involving vessels that deviate in size will be harder to find. If this is the case, it would be beneficial to include such attributes in the normal model. When more attributes that are mapped into atomic states are included, the complexity of the normal model increases which could lead to a decrease in performance. This aspect will also be evaluated.

Kinematic States Only

For the experiments with only kinematic data, the type, size and time of day are omitted. The results reveal a dramatic decrease in the detection of large vessels and wrong type anomalies, compared to the results in Table A.10. The decrease in the detection performance of the strange time anomalies is not that dramatic, 0% detected anomalies compared to 8% (sliding window approach) and 7% (precise approach), respectively. The strange time anomalies are hard to detect regardless which setup is used, which could be due to the way the strange time anomalies are generated and the time attribute discretisation. When a strange time anomaly is added to a track, it may, for example, result in moving the track from the morning to the afternoon, and this track might be as normal as before.

The total precision for the sliding window approach is slightly higher when only kinematic data is used, but the recall is lower for both methods. The biggest problem with the kinematic-only model is that no alarm settings for more than 1% of alarms can be found, this way, the maximum number of anomalies that can be detected is lower compared to when the size, time and type informa-

tion is used. Furthermore, this experiment shows the importance of modelling contextual information.

No Time Information

Omitting the time information from the experiments results in a decrease in the detection of *strange time* anomalies; only 1–2% of the anomalies are detected, compared to 8% (sliding windows) and 7% (precise approach) when all atomic states are used. The reason for this is the same as described in the kinematic states only analysis. The total precision for both detectors with 5% alarm level is a little better and the recall is somewhat worse compared to the experiments when all atomic states are used. For the 1% threshold, both the precision and the recall are higher compared to the precision and the recall in the all states experiments.

No Type Information

When the type information is omitted, a dramatic decrease in *wrong type* anomalies can be seen. Only 1–4% of the anomalies are detected compared to 69–85% when all composite states are used. This is not very surprising, considering the definition of a *wrong type* anomaly. Without type information, the detectors cannot differentiate between a normal vessel and one with the same behaviour but of another type. The 1–4% anomalies that are found even though the type information is omitted are most likely the result of a deviation in some of the other attributes. Both the precision and recall for both alarm levels were worse than using all composite states.

No Size Information

The experiment where the size information is omitted follows the same pattern as previous composite state experiments. Although, the detection rate of *large vessel* anomalies decreases from 52–87% to 11–34%, the decrease is not as significant as for the experiments without type information. At the 5% alarm level the precision is slightly better and the recall is worse, compared to the all atomic states experiments. For the 1% level, the precision is worse and the recall is slightly better.

No Position Information

The detection performance, when position information is not used is on par with the all states experiments. At the 5% alarm level, the precision is slightly higher and the recall lower, compared to the all states experiments. For the 1% alarm level, both the precision and the recall are better without position information, compared to using all atomic states. It is especially the strange manoeuvres and the high/low speed anomalies that are easier to find. This is

due to the parameter settings; for the sliding window approach the threshold and window size is [5, 5] in the 1% experiments without position. In the all atomic states experiments, the corresponding thresholds are [75, 75]. The latter thresholds clearly make the anomalies of length 50 harder to find. The same behaviour can be observed for the precise approach where the thresholds are [11, 11] without positional information and [81, 121] with all atomic states. At the 1% alarm level, the normal model without positional information should be used instead of the one with a 3 by 3 positional grid.

6.6.6 Temporal Detection Method

The experiments demonstrate that the difference between the sliding window approach and the precise approach is quite small. This could be expected, due to the fact that both methods use the same "base" detectors, and when the parameter search method for finding thresholds that result in a fixed rate of alarms is used, both methods are expected to perform roughly the same. There are, however, differences between the methods; Table 6.18 shows the best results at each alarm level, i.e., those results from the 3 by 3 grid and all atomic states for the 10% and 5% levels and the results of the experiments without position for the 1% level.

At the 10% alarm level, the sliding window method is just slightly better than the precise approach, the only difference being one more false alarm for the precise approach. At 5%, the sliding window method is better since it finds 19 more anomalies for the same number of false alarms, while at the 1% level, it is harder to find a winner, the precise approach detects more anomalies, but has twice the number of false alarms. In a real-world application, the number of false alarms is probably more important than the recall and ,therefore, the sliding window approach might be a better choice. Note that this conclusion is only true in this setup, when the parameter search method to find thresholds is used. In a setup where the operator determines the thresholds, the precise approach is more intuitive and might increase the possibility of setting suitable thresholds.

6.6.7 Highest Ranked Anomalies

As can be seen in Figure 6.8 on page 151, the false alarms are sparsely distributed among the correctly classified anomalies. This indicates that even at a higher false alarm rate, the impact of the false alarms can be reduced by presenting the anomalies in a *top list*. The data presented in the figure is, however, a batch of five days of anomalies and the top list at a specific time instance during the five days might look different.

Table 6.18: The best results of all experiments for a given alarm level.

Alarm threshold	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
10%	Sliding Window, 3 by 3 grid	420	538	66	184	0.86420	0.69536	10.93%
10%	Precise, 3 by 3 grid	420	537	67	184	0.86242	0.69536	11.09%
5%	Sliding Window, 3 by 3 grid	401	568	36	203	0.91762	0.66391	5.96%
5%	Precise, 3 by 3 grid	391	568	36	213	0.91569	0.64735	5.96%
1%	Sliding Window, no position	244	601	3	360	0.98785	0.40397	0.50%
1%	Precise, no position	255	598	6	349	0.97701	0.42218	0.99%

6.6.8 Class Imbalance Problem

In all the experiments, the evaluation dataset consists of 50% normal vessels and 50% anomalies. However, in a real-world scenario, such a distribution of the data is unlikely. This problem is referred to as the *class imbalance problem* and is discussed in the background chapter. An important question to ask is: What happens if there are fewer anomalous that normal vessels? In a real-world scenario, the anomalous vessels are very few compared to the normal ones. Assume that we have just 6 anomalies in the data instead of 604. How does this affect the performance of the method? First, it does not affect the number of false alarms. It will be the same, assuming we have 604 normal vessels in the evaluation dataset. If the 6 anomalies are a representative subset of the 604 in the original evaluation dataset, the detection performance (recall) should be at the same level. This means that the method could detect 66% of the anomalies at the 5% level with the sliding window approach. This translates into about four out of six detected anomalies. Given that there are 36 false alarms with this configuration, the precision would be 0.1, which is a very low number. However, the system should still be useful, because one out of ten alarms corresponds to an anomaly. At the 1% level, the numbers are slightly different. Two out of six anomalies would be found along with three false alarms, which results in a precision of 0.4 that is, four times the precision at the 5% level. Moreover, having 40% of all alarms translate to a found anomaly might be very useful for an operator.

6.6.9 Feasibility of the Method in a Real-World System

To assess the method's feasibility for real-world applications, the results can be mapped into the number of times false alarms are presented to the operator. In the source dataset there are about 340 vessels per day. Using the sliding window results in Table 6.18, the following number of alarms per time unit can be calculated:

- 0.5% of 340 $\approx$ 2 false alarms / day $\approx$ 0.08 false alarms / hour $\approx$ 1 false alarm / 12 hours .
- 6% of 340 $\approx$ 21 false alarms / day $\approx$ 0.9 false alarms / hour.
- 11% of 340 $\approx$ 37 false alarms / day $\approx$ 1.4 false alarms / hour.

These numbers were presented to a number of SMEs at a maritime security workshop. Their comments were that even at the 11% alarm threshold, the level of false alarms was not considered to be a major problem for the operator. At a VTS centre, like the one described in Section 6.4.2, the SMEs argued that the work of the operator could sometimes be very tiresome, because nothing happens most of the time. Therefore, an occasional alarm now and then could in fact help keeping the operator *in the loop*, and maintain focus on the task

at hand, even if the alarm does not correspond to an actual anomaly. When a system such as automatic anomaly detection is used, there is always a risk that the operators miss true anoamlies while investigating alerts from the automatic system. However, this risk should be weighed against the benefits of such a system.

Another important aspect regarding the false alarms is that in the evaluation procedure they are called false alarms and considered to be an extra burden on the operator. But in reality, this is only partly valid. A closer analysis of some of the normal vessels classified as anomalies by the method revealed that some of the vessels in fact behaved quite strangely and the operator should be alerted, to investigate and determine whether they really are anomalies. Seibert et al. [121] argue it is better that the operators only assess detected behaviours rather than having them both detect and assess behaviours.

6.7 Summary and Conclusions

We have extended the State-Based Anomaly Detection approach by introducing a precise anomaly detector using the Bayesian combination operator. A number of experiments were performed with two different datasets; the first included synthetic anomalies and the experiments measured detection delay, precision and recall. The results of the detection delay experiment showed that our precise detector outperforms previous detectors based on Gaussian mixture model and kernel density functions. The precise detector performed slightly better than the previous sliding window detector, while the results of the precision and recall experiment demonstrated that the precise detector is better than the sliding window detector in two cases out of three (threshold setups B and C with $T_{lev} = 5$ and $T_{lev} = 10$, respectively). The recall for all the detectors was low without increasing the sensitivity using the threshold with the exception of the new version of the sliding window detector, by using the max operator with the occurrence and transition detectors, it performs very well with threshold setup A (with $T_{lev} = 1$) and there is no need to increase the sensitivity of the detector. One benefit of utilizing the precise anomaly detector, instead of the sliding window detector, is that setting the reactivity of a detector by determining the number of extreme observations needed to raise an alarm can be more intuitive, compared to setting a window size and a detection threshold for a number of binary detectors.

The second dataset was much larger than the first one and the anomalies that were used were developed together with subject matter experts (SMEs) to reflect those likely to occur in a real-world scenario detecting them would be of interest for the task of the SMEs. A number of aspects of the SBAD method were evaluated: the temporal detection method, i.e., whether the sliding window or the precise approach performs best, different fusion schemes, how the length of anomaly and size of position grid affects the detection performance, how the choice of atomic states influences the types of anomalies that can be de-

tected, the computational complexity and how the false alarms are distributed compared to the detected anomalies. The conclusions from the experiments are that the max operator with the occurrence and transition detectors is the most suitable way of fusing the output from the detectors. In addition, the sliding window approach performed better than the precise approach, when the final and best results for each alarm level were combined. The length of the anomaly did not affect the performance as much as could be expected and a grid size of 3 by 3 performed best for the 5% and 10% alarm levels. For the 1% alarm level, a normal model without positional information performed best. Removing context attributes such as type, time and size from the normal model usually led to a decrease in the detection performance for anomalies involving the removed attributes. The computational cost of the method is very low and could be reduced even further by some optimizations. The top list experiments revealed that the false alarms do not interfere with the anomalies at the top of the list. After analysing the feasibility of using the method in a real-world system the SMEs maintain that the system could aid an operator, even with a false alarm level up to 11%. If the parameters are automatically set with respect to the number of false alarms, the sliding window approach should be used. But if the parameters are to be set manually by an operator, the precise approach allows for a more intuitive parameters.

Chapter 7
Conclusions and Future Work

The work performed and reported on this thesis focuses on the design, implementation and evaluation of anomaly detection methods to be used in the surveillance domain. This is motivated by the fact that the amount of data collected from sensors in the domain can be vast, which can make it difficult for a decision maker to assimilate enough information to be able to make the correct decisions.

This chapter summarizes the results and contributions of the work presented in the thesis and discusses a number of limitations and possible future research directions.

7.1 Contributions

7.1.1 Research Aim

In many surveillance systems, the amount of available information is huge and the manual analysis of this information is a difficult and tiresome task. The purpose of this research is to develop automatic support systems that can aid the operators in their task of analysing surveillance information. The focus is data-driven anomaly detection and the aim of the thesis and the research, as defined in Section 1.1, is to:

> Develop an accurate, data-driven anomaly detection method that is transparent, computationally efficient, can incorporate contextual information and expert knowledge, can handle joint features and use time-series information to detect anomalies in the surveillance domain.

The aim defines a number of important properties for a data-driven anomaly detection method. The accuracy property refers to the detection performance of the method. A good method should correctly classify anomalous tracks as anomalous and at the same time avoid classifying normal tracks as anomalous. The computational efficiency property relates to the fact that surveillance

systems are used with sensors that constantly feed new information into the system. An anomaly detection method must be able to both detect anomalies in real-time and rebuild the normal model efficiently, when the state of the world changes. The operators of surveillance systems have a deep knowledge about their domain and, if this knowledge can be used in the anomaly detection, the accuracy can potentially be increased, the operators' confidence in the system may improve, and the computational efficiency might be better. The transparency property also relates to the operators' confidence in the system, that is, being able to understand what caused an anomaly alarm relates to how much the operator trusts the system. The handling of joint features is another important property in the surveillance domain. In order to be able to capture details in the information, multiple attributes must be joint. For example, the normal speed of vehicles in a country might not be very useful in isolation, compared to whether the speed is combinded with the latitude and longitude positions or a specific road segment. The information from the sensors can be regarded as a time-series, where the behaviour of objects can be considered to be the change in attributes over time.

7.1.2 Research Objectives

To fulfill the aim, a number of research objectives were defined. The contributions in this thesis are related to the research objectives presented in Section 1.1.

O1. Characterise the properties of the surveillance domain that are important to anomaly detection.

O2. Review and analyse existing methods for anomaly detection in the surveillance domain.

O3. Propose and implement an anomaly detection method based on the result of the literature review.

O4. Evaluate the proposed method on datasets from different areas of the surveillance domain.

O5. Compare the proposed method to other methods previously used in the surveillance domain.

Characterise the Properties of the Surveillance Domain of That are Important to Anomaly Detection

The surveillance domain can be divided into a number of sub-areas, such as maritime, road, air and public areas. Anomaly detection has been identified as an enabling technique for achieving situation awareness in the surveillance

domain. The most important properties of the surveillance domain identified in the literature review in Section 3.1 are:

- A large number of objects are present in the area-of-interest.
- The area-of-interest is surveyed by a number of interconnected, sometimes heterogeneous, sensors and humans.
- Sensors provide kinematic data, but no information about relations between objects, or relations between objects and the environment.
- Sensors cannot automatically label observed objects as normal or anomalous, i.e., recorded datasets are unlabelled.
- There is a great deal of contextual information.
- Information from sensors is both temporal and spatial.
- Objects have few attributes compared, for example, to the network intrusion domain.
- The real-time aspect is very important.

Based on these properties, anomaly detection methods tailored to other domains cannot be easily used without significant changes in the pre-processing, representation and normal modelling. The use of contextual information and the use of spatio-temporal datasets are especially fundamental differences between the surveillance domain and others such as the network intrusion domain.

Review and Analyse Existing Methods for Anomaly Detection in the Surveillance Domain

In order to be able to assess other approaches for anomaly detection, a number of quantitative and qualitative properties were identified (see Section 3.2). The quantitative properties were used in the empirical evaluations of anomaly detection methods. The following quantitative properties were identified as most important: precision (the number of found anomalies that are real ones), recall (the number of anomalous objects found) and computational cost (the time it takes to build the normal model and to detect anomalies in new data).

A number of qualitative properties were also identified. These properties were used to assess other anomaly detection approaches in the literature review, presented in Section 3.3. The following properties were identified: the handling of contextual information, transparency, the incorporation of expert knowledge, robustness, adaptability and the handling of joint attributes. Seven previous approaches for anomaly detections were reviewed and subjectively assessed on the basis of the six qualitative properties.

The result of the literature review (Section 3.4) revealed that most approaches had problems handling contextual information and incorporating expert knowledge. Some approaches also had difficulties with joint attributes and that the normal models could not easily be adapted, when the normal behaviour changed in the world. This result contributes to the requirements on future anomaly detection methods, which should fulfil these properties to a much higher degree, than the methods analysed in the literature review.

Another important contribution of the literature review is a theoretical argumentation, supported by a small empirical study, which concludes that methods based on distance metrics, such as the Euclidian distance, are not appropriate to use together with datasets such as the AIS dataset (see Section 3.3.9).

Propose and Implement an Anomaly Detection Method Based on the Result of the Literature Review

Based on the result of the literature review (see Section 3.4) together with the background section on different anomaly detection methods (see Section 2.2), the State-Based Anomaly Detection (SBAD) method was proposed in Section 4.1. The proposed method employs a discrete state-based representation for both contextual information and real-time data from sensors.

The method has very low computational requirements, both when building the normal model and when detecting anomalies. The state-based representation is built on a number of state classes. A state class is a discretisation of an attribute, e.g., heading, position or velocity.

A state class encompasses a number of possible states. For example, the class CourseState might have four different states: north, south, east and west. The bounds for each state are set by a domain expert on the basis of what can be expected in the dataset from the given domain. This enables the incorporation of expert knowledge and makes the representation more transparent to a human operator. CourseState is an example of an atomic state class, which can be combined with other atomic state classes into a composite state class. This way, joint features can be modelled in the normal model.

The normal model is built using the composite states in three different ways. First, the relative frequency of all composite states in the training data is used to obtain the probability of a new observation being normal. Next, the probability of transitions between the composite states is used to find anomalous transitions between new observations. Finally, the time objects stay in the same composite state in the training data is used to find objects that stay in a composite state for an anomalously long duration. The results of these three classifiers are fused together to obtain a total degree of anomaly for the observed object.

The possibility of using expert knowledge to set the states and boundaries between states makes the model easy to adapt for a specific domain. The representation also includes a method for combining joint features, and enables fast construction of normal models, as well as the possibility of updating parts of

the normal model without rebuilding it from scratch. The three classifiers can detect a wide range of different classes of anomalies.

Evaluate the Proposed Method on Datasets From Different Areas of the Surveillance Domain

The SBAD method has been evaluated on a number of datasets from different areas in the surveillance domain, such as indoor (Section 5.2) and outdoor (Section 4.1) video surveillance, maritime surveillance (Section 4.2 and Chapter 6) and ground transportation (Section 5.1). The SBAD method performs very well in most of these domains except, for the area of indoor video surveillance.

In the outdoor video surveillance experiments, the SBAD method was able to find simple, single object anomalies and more complex, multi-object ones, without too many false alarms. In these experiments, the use of a map for context information and information about relations between objects enables the normal model to be quite simple and, yet, maintain a high level of classification performance. Both maritime experiments show that the SBAD method is very well suited for this kind of surveillance. The behaviour of maritime vessels is not as erratic as behaviour in the indoor video surveillance domain and, it is therefore much easier to build a model of normal behaviour. The SBAD method was able to detect most of the evaluated maritime anomalies, without too many false alarms. The ground transportation area is another example of where the SBAD method excels in performance. The commuter experiments demonstrate that the SBAD method is very adept at learning the normal behaviour of objects travelling along roads. Although the dataset was quite small, the experiments show similar results. In the area of indoor video surveillance, the number of false alarms was quite high. While, this was mainly a result of the complexity of the scene, it was also related to problems with the lower level fusion systems (tracking and correlation). Nevertheless, despite the large number of false alarms, the SBAD method performed better than the method based on Gaussian Mixture Models (GMM).

In order to be able to evaluate whether the SBAD method is applicable in an operative real-world system, an evaluation dataset was developed together with subject matter experts (see Section 6.4). The dataset included a number of instances from eight different classes of anomalies, which were derived from a litterature review and interviews with subject matter experts. The identified anomaly classes were both considered to be likely to occur in the real world and detecting them was considered interesting to the Swedish Maritime Administration (Sjöfartsverket). The result of the evaluation was presented to the subject matter experts and they indicated that the number of detections and false alarms were well inside the boundaries of what could be considered to be useful in an operative system (see Section 6.6.9).

Compare the Proposed Method to Other Methods Previously Used in the Surveillance Domain

The proposed SBAD method has been evaluated and compared with two other popular approaches for anomaly detection in the surveillance domain, namely GMM and KDE. Moreover, the evaluation was performed on the same datasets with the same conditions. The GMM and KDE methods were implemented by the co-author of Brax et al. [28, 30]. The result of the evaluations was that the SBAD method performed much better than the GMM and KDE methods in the maritime area (see Section 6.3). Furthermore, the SBAD also performed better than the GMM in the indoor video surveillance experiment (see Section 5.2) and as well as in the outdoor video surveillance experiment (see Section 4.1).

7.1.3 Results Related to Research Aim

The aim of this work was to develop an anomaly detection method that had a number of important properties. In this section, the SBAD method is analysed with respect to the thesis aim, to assess the degree to which the aim can be considered fulfilled.

Based on the experiments, the proposed method, SBAD, can be considered to be more accurate than the GMM and KDE methods. Its computational efficiency is very high compared to the GMM and KDE methods. Moreover, the computational cost is low enough to enable the normal model to be built in real-time given that the number of updates is fewer than 30.000 per second (see Section 6.5.1). Anomaly detection is less expensive than building the normal model and can be done in real-time with up to 200.000 updates per second on the computer used in the experiments in Section 6.4. These numbers are derived from experiments where object updates were received from a file. If the object updates are received from a network interface, the number of processed updates per second can be lower, due to overhead in the network stack. As shown in all the experiments, incorporating various kinds of contextual information into the SBAD normal model is very easy and is done by adding new atomic state classes to the composite state class used when building the normal model. Examples of contextual information used in the experiments are: time of day, maps, vessel type and relations between objects. Expert knowledge in the form of parameter settings for the discretisation process and parameters for the detection of anomalies has been included in the method. For example, the different size classes and their boundaries used in the experiments in Section 6.4 were based on the SMEs' knowledge. The transparency of the model has not been extensively evaluated, but the operator can easily obtain information about which composite states are classified as anomalous on a track. An anomalous composite state may appear as: $C_n = \{north, fast, pos25, night\}$, which corresponds to an object moving north at high speed in a specific area at night. Joint features in the SBAD method are handled by the composite state

class. In this way, it is very easy to add more features to make a more finely-grained normal model, e.g., adding the time of day to a composite state will make it possible to model daily variations in an area of interest. The data from the sensors is multi-dimensional time-series data, and the time aspect of this data is handled in the SBAD method by the sliding window and precise detectors introduced in Section 5.1 and Chapter 6 respectively. This enables the SBAD method to detect both small anomalies that develop over time as well as large ones occurring more or less momentarily. The normal model can be adapted to change the definition of normality, which is done by just adding or removing the number of instances of each composite state occurrence or transition.

Based on this analysis, it can be said that the aim of the thesis has been fulfilled to a high degree, with the exception of the transparency property, which was not extensively evaluated. The SBAD method fulfills the aim better than any of the previous approaches analysed in Section 3.3.

To summarize, the main contributions are (1) a new method for anomaly detection called *The State-Based Anomaly Detection* method, (2) an evaluation of the method on a number of surveillance datasets, (3) a performance comparison between the SBAD method and two approaches based on GMM and KDE, (4) an extension to the SBAD method for detecting anomalous state transitions and anomalous stops and (5) a study of real-world anomalies for the evaluation of anomaly detection methods, as well as a feasibility assessment of the use of the SBAD method in the maritime domain.

The SBAD method is suitable for anomaly detection problems where: contextual information is important, the distribution of attributes in the data is unknown, domain knowledge is available, the normal model has to be updated online, the data includes both discrete and continious attributes and the computatinal cost of building the normal model is important. The SBAD method is not suitable for problems where the anomalies can be in the form of the absense of data, i.e., no traffic on a road which normally has dense traffic.

7.2 Future Work

During the work with the SBAD method, a number of shortcomings and ideas about future development of the method were identified. Since time was a limiting factor, most of those ideas could not be investigated further. This section elaborate on some of those ideas.

7.2.1 Other Domains

In this work, the SBAD method was evaluated on a number of different datasets, all of which represented aspects of the surveillance domain. Physical objects are surveyed with sensors in this domain and, therefore, the main features in the datasets are kinematic, e.g., features related to the motion of objects.

Examples of such features include latitude and longitude position, speed and course.

It should be possible to use the SBAD method in other domains as well. One such domain, which has been discussed in various contexts, is transportation security based on non-kinematic data. The idea is to aid customs authorities in finding smuggling of illegal goods. This can be done by automatic analysis of cargo manifests. The atomic state classes in this context could be time-stamp, harbour id, vessel id and type of goods. Using a dataset consisting of old cargo manifests, a model of normal types of freights, could be built. With this model, the probability of transitions such as "vessel X transports a cargo of Y from harbour A to B" could be assessed and anomalous transitions could be reported.

Customs authorities are currently using a similar approach, but the task is being done by manually searching cargo manifests for strange cargo. This is very tiresome work and it is impossible to manually analyse all the manifests. According to a customs agent, the work is like trying to *find a needle in a haystack*. However, with some help of luck, strange cargo is being found now and then.

There is one important area of the surveillance domain where the SBAD method has not been evaluated, namely, air surveillance. Besides information from radar systems there is a system called *Automatic Dependent Surveillance-Broadcast* (ADS-B), which is the commercial aircraft counterpart to AIS in the maritime domain. In the end of the thesis work a large ADS-B dataset, recorded in Sweden, became available, but unfortunately, there was no time for additional experiments.

7.2.2 Normal Model Management

In all the experiments conducted in this work, it is assumed that the normal dataset is fixed, i.e., the normal data does not change. This is of course a simplification; the real world is in constant change and the normal model should be updated to reflect the state of the world. When building an operative system, the life-cycle management of the normal model is an important issue that must be addressed. Normal model management is highly dependent on the requirements of the specific domain and includes tasks such as:

- Addition of new information to the normal model.
- Removal of old, out-dated information from the normal model.
- Change of parameters of the atomic and composite states.
- Detection of deteriorating classification performance.

Other issues that are important to consider, when building an operative system, include who is responsible and/or authorized to update the normal model and how to ensure the detection performance when the normal model is updated.

Riveiro et al. [112] suggest that the operator should have an active role in maintaining the performance of the system. This may be enough for some application domains, but there should also be automatic methods for the detection of performance degradation in the normal model.

The SBAD method is designed to enable easy updating of the normal model. Adding and removing information is achieved by simply changing the frequency of observed composite states and composite state transitions. It is also possible to set a time-stamp on each composite state that is added to the normal model, which means that old information can be automatically removed from the model. Due to the very low computational complexity of the SBAD method, it is also possible to rebuild the model from scratch in a short period of time (compared to methods such as Gaussian Mixture Models built with the Expectation-Maximization algorithm). It may be necessary to rebuild the model when the parameters of the atomic or composite states are changed.

Another aspect of the normal model management is how to make the system work from day one, i.e., when no or very little training data is available. The SBAD method can be used in an online learning mode, where the model starts without any information and is built incrementally as new data begins to arrive. The normal model can be used for classification, regardless of how much training data has been fed into it. However, the classification performance will not be very good at the beginning of an incrementally built model. This issue relates to the those about how to set the atomic state boundaries from data.

7.2.3 New Atomic State Classes

A total of eight different atomic state classes were introduced into the experiments presented in this thesis. These state classes were aimed at solving the problem of short-time anomaly detection, i.e., detecting anomalies in a time span of seconds. Examples of other potentially interesting atomic state classes are ones based on percentage change of an attribute in a time interval (e.g. speed change state, with state classes -10%, 0%, 10%) and on acceleration. It would also be interesting to improve the relational state class to includes other parameters than the distance between objects. To detect anomalies over greater time spans, atomic state classes, such as, number of turns per time unit, number of accelerations per time unit and probability for travelling between different harbours, can be incorporated.

7.2.4 Behaviour Clustering/Classification

The composite states in the SBAD method have been used for anomaly detection in all the experiments. However, a sequence of composite states is actually a high-level description of object behaviour. This description could be used for a number of different tasks. One interesting task is to cluster objects on the basis of their behaviour. The clusters could be used to classify objects in real-time

and present the classification to an operator. Examples of clusters in the maritime domain are *leaving harbour, entering harbour, fishing, piloting* and so on. The clusters could also be used to find anomalies based on the long term behaviour of objects (compared to the short time behaviour that is the focus of this work). Due to the nature of the representation, a simple clustering algorithm could be to use set-theory. A specific behaviour of an object can be described as a set of composite states (which is generated from the observations of the object). All subsets and partial subsets of this set are variants of the behaviour. By identifying the size of the intersection between composite state sets from different objects, similar objects could be grouped together.

7.2.5 State Boundaries from Data

In all the experiments in this thesis, the number of states in each atomic state class, and as the boundaries between each state, are set manually. This way, some domain knowledge can be included into the normal model. This might not however, be the best way of dividing the data. Another approach is to use automatic methods to set the states and the boundaries. Such methods can be unsupervised or supervised. According to Liu et al. [94], unsupervised methods, such as equal width or equal frequency, perform well, if the data is uniformly distributed and relatively free of outliers. To overcome these problems, supervised discretisation, where the class label guides the discretisation can be used. In anomaly detection, the class label is usually not available. Another way of creating state boundaries is to use a one-dimensional clustering algorithm to find "good" clusters around which to form state classes.

The position state class is slightly different than the other state classes. This is because it is constructed from two attributes, latitude and longitude, and, hence, requires a different discretisation method. As the previous experiments show, it can be very important to set a positional grid that matches the amount of data available to build the normal model. In the experiment, a homogeneous grid was used, which may lead to problems because the amount of available data is large in some areas, but very small in others. It would be suitable to have a position grid that has different sizes in different areas, depending on how much data is available. Fortunately, there are a number of methods for generating such grids. These methods are referred to as *Spatial Indexing Methods* and are often used to create indexes of spatial data.. Samet [120] describes the PM1 Line QuadTree, that automatically splits grid cells, when the amount of data increases, and should, therefore, be suitable for creating a dynamic position state class that automatically generates a finer grid where much data exists.

7.2.6 Man Machine Interface and User Interaction

The user's interaction with an anomaly detection system is an important issue [112]. Operators often possess much knowledge about the situation at hand

and this knowledge should be used as effectively as possible, to get the most out of the system as a whole. For example, if an anomaly detection system classifies an object's behaviour as anomalous, the system cannot know whether it really is an anomaly or just something not included in the normal model. If the system classifies an object's behaviour as anomalous and the operator knows it is normal, the operator should be able to reclassify the behaviour as normal and the system should not alert the next time it encounters that behaviour. This should be true the reverse as well. If the operator finds an object with anomalous behaviour it should be possible to teach the system about this behaviour, so the behaviour can be detected the next time it occurs.

In this work, a high performance method for anomaly detection was introduced. However, the method does not provide a way of presenting the result to a human operator, which is a very important aspect of anomaly detection that should be further analysed. In some experiments, visualisation approaches, such as colour highlighting and ranked lists, were used, but not evaluated together with actual operators.

Besides using the normal model for classifying new observations it can also be used to show for the operator how the data is distributed. For example, in Figure 7.1, the spatial distribution of observations in the normal model is plotted over the map. When the number of observations in a cell increases, the colour in that cell will change from green to orange and finally to red. This kind of visualisation could also be made with other atomic states.

The statistics in the normal model could also be used to answer questions such as: "What is the probability of a person walking on the grass, given that the speed is *medium* and the course is *east*?" or "What is the probability of an object moving through a specific grid cell given that there are other objects in the vicinity?" The probabilities in the examples above can be calculated using Bayes rule and the normal model.

7.2.7 Anomaly Prediction

In the experiments, the SBAD method was used to find anomalies in real-time. However, for some applications, it might be too late to detect the anomalies when they have already occurred. It would be much better if the method could predict potential anomalies. Even if there is no direct support for this in SBAD it could easily be accomplished, by just projecting the input data into the future. For example, a dead-reckoning algorithm could be used to calculate the future position of an object, based on its history.

7.2.8 Individual Normal Models

The normal models built in this work model global normalcy of tracks in the training dataset. When, for example, AIS data is used, each vessel has a unique ID. It is therefore possible to build individual normal models and detect a single

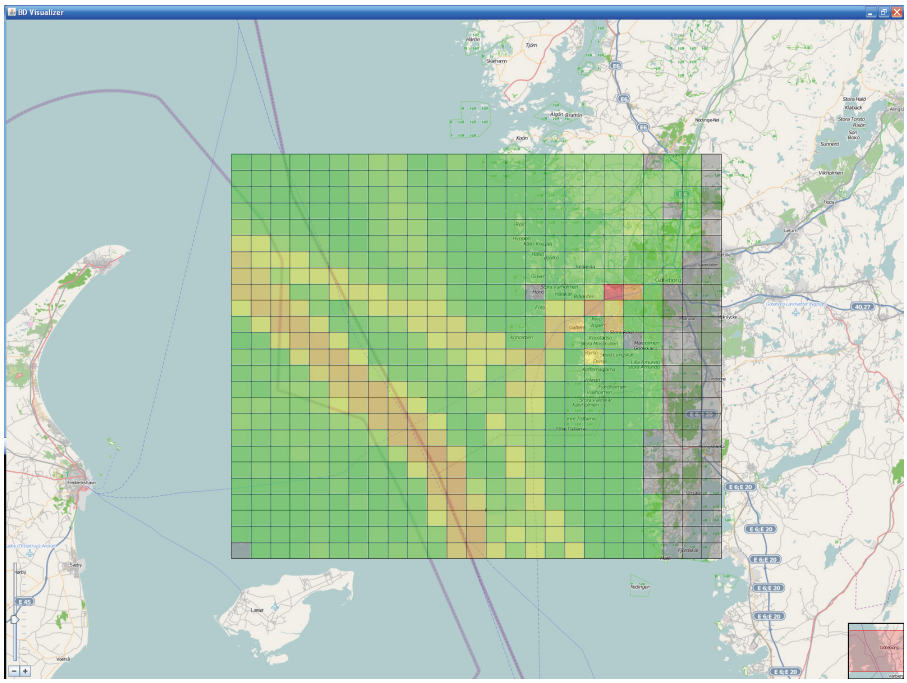


Figure 7.1: The relative frequency of observations in grid cells drawn on a map over the area used in the experiments with real-world maritime anomalies. Map from www.openstreetmap.org, © OpenStreetMap contributors, CC-BY-SA.

vessel that deviates from what is normal behaviour for that specific vessel. Applied to the transportation domain, models of individual trucks can be created and anomalies, such as, drunk drivers or engine problems could potentially be detected.

7.2.9 Multiple Composite State Classes in Parallel

In all the experiments, only one composite state class has been used. This composite state class often included as many atomic state classes as possible. For some domains it may be better to use a number of composite state classes with different subsets of atomic state classes to capture different aspects. For example, one composite state class might include position, speed and course, while another might include position and time of day. The first composite state class detects kinematic anomalies, while the other detects anomalies, such as, travelling in areas at a time when objects do not usually frequent that area. The output from the different models of normalcy should be fused together before being presented to an operator.

Appendix A
Precise SBAD: Experimental Settings and Detailed Results

This appendix holds the experimental settings and detailed results for the precise SBAD experiments with real-world anomalies.

A.1 Settings and Results

A.1.1 Fusion Scheme Experiments

Table A.1: Parameters for the fusion scheme experiments.

Fusion scheme experiments, anomaly length 50, 3x3 position grid						
Alarm threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
5%	Sliding window	4.74%	5	7	N/A	N/A
5%	Precise	4.62%	N/A	N/A	11	11

Table A.2: Results of the fusion scheme experiments.

Fusion scheme name	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
Max: Transition, Occurrence	Sliding Window	409	568	36	195	0.922	0.677	5.96%
Max: Trans., Occ., SameState	Sliding Window	604	0	604	0	0.500	1	100%
Only Transition	Sliding Window	409	568	36	195	0.920	0.677	5.96%
Only Occurrence	Sliding Window	307	578	26	297	0.922	0.508	4.30%
Only SameState	Sliding Window	448	2	602	156	0.427	0.742	99.67%
Weighted sum: Occ., Trans., SameState	Sliding Window	281	580	24	323	0.921	0.465	3.97%
Weighted sum: Occurrence, Transition	Sliding Window	328	578	26	276	0.927	0.543	4.30%
Max: Trans., Occ.	Precise	409	568	36	195	0.919	0.677	5.96%
Only Transition	Precise	399	568	36	205	0.917	0.661	5.96%
Only Occurrence	Precise	283	581	23	321	0.925	0.469	3.81%

A.1.2 Anomaly Length Experiments

Table A.3: Parameters for the anomaly length experiments.

Anomaly length experiments, 12x12 position grid						
Alarm threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
10%	Sliding window	10.03%	35	65	N/A	N/A
10%	Precise	9.24%	N/A	N/A	31	61
5%	Sliding window	5.07%	55	55	N/A	N/A
5%	Precise	4.96%	N/A	N/A	81	111
1%	Sliding window	1.13%	195	195	N/A	N/A
1%	Precise	1.01%	N/A	N/A	221	241

Table A.4: Results of the anomaly length experiments with lengths 25 and 50.

Alarm thre-shold	Method	Anomaly length	TP	TN	FP	FN	Precision	Recall	False alarm rate
10%	Sliding window	25	223	546	58	381	0.794	0.369	9.60%
10%	Precise	25	229	545	59	375	0.795	0.379	9.77%
5%	Sliding window	25	192	567	37	412	0.838	0.318	6.13%
5%	Precise	25	178	570	34	426	0.840	0.295	5.63%
1%	Sliding window	25	107	594	10	497	0.915	0.177	1.66%
1%	Precise	25	109	594	11	496	0.908	0.180	1.82%
10%	Sliding window	50	323	546	58	281	0.848	0.535	9.60%
10%	Precise	50	302	545	59	302	0.837	0.500	9.77%
5%	Sliding window	50	198	567	37	406	0.843	0.328	6.13%
5%	Precise	50	185	570	34	419	0.845	0.306	5.63%
1%	Sliding window	50	107	594	10	497	0.915	0.177	1.66%
1%	Precise	50	109	593	11	495	0.908	0.180	1.82%

Table A.5: Results of the anomaly length experiments with lengths 100 and 200.

Alarm thre-shold	Method	Anomaly length	TP	TN	FP	FN	Precision	Recall	False alarm rate
10%	Sliding window	100	327	546	58	277	0.849	0.541	9.60%
10%	Precise	100	324	545	59	289	0.846	0.529	9.77%
5%	Sliding window	100	279	567	37	325	0.883	0.462	6.13%
5%	Precise	100	245	570	34	359	0.878	0.406	5.63%
1%	Sliding window	100	110	594	10	494	0.917	0.182	1.66%
1%	Precise	100	114	593	11	490	0.912	0.189	1.82%
10%	Sliding window	200	341	546	58	263	0.855	0.565	9.60%
10%	Precise	200	344	545	59	260	0.854	0.570	9.77%
5%	Sliding window	200	284	567	37	320	0.885	0.470	6.13%
5%	Precise	200	269	570	34	335	0.888	0.445	5.63%
1%	Sliding window	200	185	594	10	419	0.949	0.306	1.66%
1%	Precise	200	128	593	11	476	0.921	0.212	1.82%

A.1.3 Grid Size Experiments

Table A.6: Parameters for the grid size experiments with grid sizes 25 by 25 and 12 by 12.

Grid size experiments							
Alarm threshold	Method	Grid size	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
10%	Sliding window	25x25	9.70%	65	75	N/A	N/A
10%	Precise	25x25	9.81%	N/A	N/A	71	141
5%	Sliding window	25x25	4.74%	105	105	N/A	N/A
5%	Precise	25x25	4.95%	N/A	N/A	251	161
1%	Sliding window	25x25	1.13%	250	250	N/A	N/A
1%	Precise	25x25	0.90%	N/A	N/A	300	300
10%	Sliding window	12x12	10.03%	35	65	N/A	N/A
10%	Precise	12x12	9.24%	N/A	N/A	31	61
5%	Sliding window	12x12	5.07%	55	55	N/A	N/A
5%	Precise	12x12	4.96%	N/A	N/A	81	111
1%	Sliding window	12x12	1.13%	195	195	N/A	N/A
1%	Precise	12x12	1.01%	N/A	N/A	221	241

Table A.7: Parameters for the grid size experiments with grid sizes 6 by 6 and 3 by 3.

Grid size experiments							
Alarm threshold	Method	Grid size	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
10%	Sliding window	6x6	9.19%	15	135	N/A	N/A
10%	Precise	6x6	9.58%	N/A	N/A	11	11
5%	Sliding window	6x6	4.51%	45	65	N/A	N/A
5%	Precise	6x6	4.62%	N/A	N/A	31	61
1%	Sliding window	6x6	1.01%	115	115	N/A	N/A
1%	Precise	6x6	1.01%	N/A	N/A	141	171
10%	Sliding window	3x3	9.02%	5	135	N/A	N/A
10%	Precise	3x3	9.47%	N/A	N/A	1	5
5%	Sliding window	3x3	4.74%	5	7	N/A	N/A
5%	Precise	3x3	4.62%	N/A	N/A	11	11
1%	Sliding window	3x3	0.90%	75	75	N/A	N/A
1%	Precise	3x3	1.01%	N/A	N/A	81	121

Table A.8: Results of the grid size experiments with grid sizes 25 by 25 and 12 by 12.

Target threshold	Method	Grid size	TP	TN	FP	FN	Precision	Recall	False alarm rate
10%	Sliding window	25x25	215	548	56	389	0.793	0.356	9.27%
10%	Precise	25x25	233	543	61	371	0.793	0.386	10.10%
5%	Sliding window	25x25	167	575	29	437	0.852	0.276	4.80%
5%	Precise	25x25	153	571	33	451	0.823	0.253	5.46%
1%	Sliding window	25x25	98	599	5	506	0.951	0.162	0.83%
1%	Precise	25x25	95	593	11	509	0.896	0.157	1.82%
10%	Sliding window	12x12	323	546	58	281	0.848	0.535	9.60%
10%	Precise	12x12	302	545	59	302	0.837	0.500	9.77%
5%	Sliding window	12x12	199	567	37	406	0.843	0.329	6.13%
5%	Precise	12x12	185	570	34	419	0.845	0.306	5.63%
1%	Sliding window	12x12	107	594	10	497	0.915	0.177	1.66%
1%	Precise	12x12	109	593	11	495	0.908	0.180	1.82%

Table A.9: Results of the grid size experiments with grid sizes 6 by 6 and 3 by 3.

Alarm thre-shold	Method	Grid size	TP	TN	FP	FN	Precision	Recall	Actual alarm rate
10%	Sliding window	6x6	418	549	55	186	0.884	0.692	9.11%
10%	Precise	6x6	422	541	63	182	0.870	0.699	10.43%
5%	Sliding window	6x6	278	574	30	326	0.903	0.460	4.97%
5%	Precise	6x6	266	571	33	338	0.890	0.440	5.46%
1%	Sliding window	6x6	130	592	12	474	0.916	0.215	1.99%
1%	Precise	6x6	129	590	14	475	0.902	0.214	2.32%
10%	Sliding window	3x3	420	538	66	184	0.864	0.695	10.93%
10%	Precise	3x3	420	537	67	184	0.862	0.695	11.09%
5%	Sliding window	3x3	401	568	36	203	0.918	0.664	5.96%
5%	Precise	3x3	391	568	36	213	0.916	0.647	5.96%
1%	Sliding window	3x3	153	597	7	451	0.956	0.253	1.16%
1%	Precise	3x3	155	597	7	449	0.957	0.257	1.16%

A.1.4 Composite State Experiments

Table A.10: Results of composite state experiments, detailed anomaly class results with a normal model using all atomic states.

Anomaly class	Alarm threshold	Method	TP	FN	Anomalies found
Circle and land	5%	Sliding Window	1	1	50%
	5%	Precise	2	0	100%
	1%	Sliding Window	0	2	0%
	1%	Precise	1	1	50%
Large vessels	5%	Sliding Window	87	13	87%
	5%	Precise	87	13	87%
	1%	Sliding Window	52	48	52%
	1%	Precise	57	43	57%
Strange man.	5%	Sliding Window	96	4	96%
	5%	Precise	93	7	93%
	1%	Sliding Window	3	97	3%
	1%	Precise	1	99	1%
Missed turn	5%	Sliding Window	1	1	50%
	5%	Precise	0	2	0%
	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
High/low speed	5%	Sliding Window	53	47	53%
	5%	Precise	47	53	47%
	1%	Sliding Window	1	99	1%
	1%	Precise	1	99	1%
Unexp. stops	5%	Sliding Window	64	36	64%
	5%	Precise	64	36	64%
	1%	Sliding Window	2	98	2%
	1%	Precise	3	97	3%
Strange time	5%	Sliding Window	15	85	15%
	5%	Precise	13	87	13%
	1%	Sliding Window	8	92	8%
	1%	Precise	7	93	7%
Wrong type	5%	Sliding Window	84	16	84%
	5%	Precise	85	15	85%
	1%	Sliding Window	69	31	69%
	1%	Precise	73	27	73%

Table A.11: Parameters for the composite state experiments with a normal model built only on kinematic atomic states. Unfortunately, it was only possible to find parameters for the 1% alarm threshold.

Composite state experiment parameters: Kinematic states						
Alarm threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
1%	Sliding window	0.68%	5	75	N/A	N/A
1%	Precise	0.68%	N/A	N/A	1	3

Table A.12: Results of the composite state experiments with only kinematic atomic states. Complete evaluation dataset.

Alarm threshold	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
1%	Sliding Window	134	603	1	470	0.993	0.222	0.17%
1%	Precise	138	599	5	466	0.965	0.229	0.83%

Table A.13: Results of composite state experiments, detailed anomaly class results of using only position, course and speed.

Anomaly class	Alarm threshold	Method	TP	FN	Anomalies found
Circle and land	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
Large vessels	1%	Sliding Window	6	94	6%
	1%	Precise	8	92	8%
Strange man.	1%	Sliding Window	96	4	96%
	1%	Precise	97	3	97%
Missed turn	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
High/low speed	1%	Sliding Window	3	97	3%
	1%	Precise	4	96	4%
Unexp. stops	1%	Sliding Window	29	71	29%
	1%	Precise	29	71	29%
Strange time	1%	Sliding Window	0	100	0%
	1%	Precise	0	100	0%
Wrong type	1%	Sliding Window	0	100	0%
	1%	Precise	0	100	0%

Table A.14: Parameters for the composite state experiments without time state.

Composite state experiment parameters: No time state						
Target threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
5%	Sliding window	2.25%	5	105	N/A	N/A
5%	Precise	2.25%	N/A	N/A	1	11
1%	Sliding window	0.79%	15	15	N/A	N/A
1%	Precise	0.90%	N/A	N/A	11	51

Table A.15: Results of the composite state experiments without time state. Complete evaluation dataset.

Alarm threshold	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
5%	Sliding Window	366	584	20	238	0.948	0.606	3.31%
5%	Precise	358	582	22	246	0.942	0.593	3.64%
1%	Sliding Window	255	595	9	349	0.966	0.422	1.49%
1%	Precise	232	594	10	372	0.959	0.384	1.66%

Table A.16: Results of composite state experiments, detailed anomaly class results for a composite state without the atomic time state.

Anomaly class	Alarm threshold	Method	TP	FN	Anomalies found
Circle and land	5%	Sliding Window	2	0	100%
	5%	Precise	1	1	50%
	1%	Sliding Window	1	1	50%
	1%	Precise	1	1	50%
Large vessels	5%	Sliding Window	86	14	86%
	5%	Precise	86	14	86%
	1%	Sliding Window	79	21	79%
	1%	Precise	78	22	78%
Strange man.	5%	Sliding Window	98	2	98%
	5%	Precise	93	7	93%
	1%	Sliding Window	23	77	23%
	1%	Precise	9	91	9%
Missed turn	5%	Sliding Window	0	2	0%
	5%	Precise	1	1	50%
	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
High/low speed	5%	Sliding Window	41	59	41%
	5%	Precise	38	62	38%
	1%	Sliding Window	21	79	21%
	1%	Precise	20	80	20%
Unexp. stops	5%	Sliding Window	55	45	55%
	5%	Precise	55	45	55%
	1%	Sliding Window	55	45	55%
	1%	Precise	45	55	45%
Strange time	5%	Sliding Window	2	98	2%
	5%	Precise	2	98	2%
	1%	Sliding Window	1	99	1%
	1%	Precise	1	99	1%
Wrong type	5%	Sliding Window	82	18	82%
	5%	Precise	82	18	82%
	1%	Sliding Window	75	25	75%
	1%	Precise	78	22	78%

Table A.17: Parameters for the composite state experiments without type state.

Composite state experiment parameters: No type state						
Alarm threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
5%	Sliding window	4.96%	5	45	N/A	N/A
5%	Precise	3.83%	N/A	N/A	1	11
1%	Sliding window	0.90%	35	145	N/A	N/A
1%	Precise	0.90%	N/A	N/A	31	41

Table A.18: Results of the composite state experiments without type state. Complete evaluation dataset.

Alarm threshold	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
5%	Sliding Window	281	567	37	323	0.884	0.465	6.13%
5%	Precise	279	570	34	325	0.891	0.462	5.96%
1%	Sliding Window	137	593	11	467	0.926	0.227	1.82%
1%	Precise	133	592	12	471	0.917	0.220	1.99%

Table A.19: Results of composite state experiments, detailed anomaly class results for a composite state without the atomic type state.

Anomaly class	Alarm threshold	Method	TP	FN	Anomalies found
Circle and land	5%	Sliding Window	2	0	100%
	5%	Precise	1	1	50%
	1%	Sliding Window	1	1	50%
	1%	Precise	1	1	50%
Large vessels	5%	Sliding Window	77	23	77%
	5%	Precise	78	22	78%
	1%	Sliding Window	54	46	54%
	1%	Precise	54	46	54%
Strange man.	5%	Sliding Window	98	2	98%
	5%	Precise	97	3	97%
	1%	Sliding Window	10	90	10%
	1%	Precise	6	94	6%
Missed turn	5%	Sliding Window	1	1	50%
	5%	Precise	1	1	50%
	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
High/low speed	5%	Sliding Window	38	62	38%
	5%	Precise	36	64	36%
	1%	Sliding Window	13	87	13%
	1%	Precise	13	87	13%
Unexp. stops	5%	Sliding Window	53	47	53%
	5%	Precise	52	48	52%
	1%	Sliding Window	52	48	52%
	1%	Precise	52	48	52%
Strange time	5%	Sliding Window	9	91	9%
	5%	Precise	10	90	10%
	1%	Sliding Window	6	94	6%
	1%	Precise	6	94	6%
Wrong type	5%	Sliding Window	3	97	3%
	5%	Precise	4	96	4%
	1%	Sliding Window	1	99	1%
	1%	Precise	1	99	1%

Table A.20: Parameters for the composite state experiments without size state.

Composite state experiment parameters: No size state						
Alarm threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
5%	Sliding window	3.27%	5	205	N/A	N/A
5%	Precise	3.16%	N/A	N/A	1	11
1%	Sliding window	0.56%	15	15	N/A	N/A
1%	Precise	0.79%	N/A	N/A	11	21

Table A.21: Results of the composite state experiments without size state. Complete evaluation dataset.

Alarm threshold	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
5%	Sliding Window	315	574	30	289	0.913	0.522	4.97%
5%	Precise	297	582	22	307	0.931	0.492	3.64%
1%	Sliding Window	188	594	10	416	0.950	0.311	1.66%
1%	Precise	208	593	11	396	0.950	0.344	1.82%

Table A.22: Results of composite state experiments, detailed anomaly class results for a composite state without the atomic size state.

Anomaly class	Alarm threshold	Method	TP	FN	Anomalies found
Circle and land	5%	Sliding Window	2	0	100%
	5%	Precise	1	1	50%
	1%	Sliding Window	1	1	50%
	1%	Precise	1	1	50%
Large vessels	5%	Sliding Window	34	66	34%
	5%	Precise	29	71	29%
	1%	Sliding Window	11	89	11%
	1%	Precise	21	79	21%
Strange man.	5%	Sliding Window	98	2	98%
	5%	Precise	95	5	95%
	1%	Sliding Window	23	77	23%
	1%	Precise	28	72	28%
Missed turn	5%	Sliding Window	1	1	50%
	5%	Precise	1	1	50%
	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
High/low speed	5%	Sliding Window	32	68	32%
	5%	Precise	25	75	25%
	1%	Sliding Window	17	83	17%
	1%	Precise	19	81	19%
Unexp. stops	5%	Sliding Window	59	41	59%
	5%	Precise	57	43	57%
	1%	Sliding Window	55	45	55%
	1%	Precise	55	45	55%
Strange time	5%	Sliding Window	8	92	8%
	5%	Precise	8	92	8%
	1%	Sliding Window	7	93	7%
	1%	Precise	7	93	7%
Wrong type	5%	Sliding Window	81	19	81%
	5%	Precise	81	19	81%
	1%	Sliding Window	74	26	74%
	1%	Precise	77	23	77%

Table A.23: Parameters for the composite state experiments without position state.

Composite state experiment parameters: No position state						
Alarm threshold	Method	Actual alarm rate	T_{al}	n_{ws}	n_{ext}^{Occ}	n_{ext}^{Tr}
5%	Sliding window	2.59%	5	155	N/A	N/A
5%	Precise	3.27%	N/A	N/A	1	3
1%	Sliding window	1.01%	5	5	N/A	N/A
1%	Precise	1.01%	N/A	N/A	11	11

Table A.24: Results of the composite state experiments without position state. Complete evaluation dataset.

Alarm threshold	Method	TP	TN	FP	FN	Precision	Recall	False alarm rate
5%	Sliding Window	291	591	13	313	0.957	0.482	2.15%
5%	Precise	294	588	16	310	0.948	0.487	2.65%
1%	Sliding Window	244	601	3	360	0.988	0.404	0.50%
1%	Precise	255	598	6	349	0.977	0.422	0.99%

Table A.25: Results of composite state experiments, detailed anomaly class results for composite states without the atomic position state.

Anomaly class	Alarm threshold	Method	TP	FN	Anomalies found
Circle and land	5%	Sliding Window	1	1	50%
	5%	Precise	1	1	50%
	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
Large vessels	5%	Sliding Window	68	32	68%
	5%	Precise	69	31	69%
	1%	Sliding Window	54	46	54%
	1%	Precise	59	41	59%
Strange man.	5%	Sliding Window	98	2	98%
	5%	Precise	98	2	98%
	1%	Sliding Window	81	19	81%
	1%	Precise	89	11	89%
Missed turn	5%	Sliding Window	0	2	0%
	5%	Precise	0	2	0%
	1%	Sliding Window	0	2	0%
	1%	Precise	0	2	0%
High/low speed	5%	Sliding Window	37	63	37%
	5%	Precise	39	61	39%
	1%	Sliding Window	32	68	32%
	1%	Precise	28	72	28%
Unexp. stops	5%	Sliding Window	5	95	5%
	5%	Precise	6	94	6%
	1%	Sliding Window	0	100	0%
	1%	Precise	0	100	0%
Strange time	5%	Sliding Window	7	93	7%
	5%	Precise	8	92	8%
	1%	Sliding Window	7	93	7%
	1%	Precise	7	93	7%
Wrong type	5%	Sliding Window	75	25	75%
	5%	Precise	73	27	73%
	1%	Sliding Window	70	30	70%
	1%	Precise	72	28	72%

Bibliography

[1] The Third International Knowledge Discovery and Data Mining Tools Competition Dataset. WWW, Accessed: July 12, 2010 . URL http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[2] IEEE Standard Computer Dictionary. A Compilation of IEEE Standard Computer Glossaries. *IEEE Std 610*, 1991.

[3] National plan to achieve maritime domain awareness. Technical report, Department of Homeland Security, USA, 2005.

[4] Maritime Traffic Information, Swedish Maritime Administration. WWW, Accessed: October 2, 2010. URL http://www.sjofartsverket.se/en/Infrastructure--Maritime-Traffic/Maritime-Traffic-Information/.

[5] University of California, Irvine, Machine Learning Repository. WWW, Accessed March 24, 2011. URL http://archive.ics.uci.edu/ml/.

[6] Erieye radar. WWW, Accessed March 9, 2009. URL http://en.wikipedia.org/wiki/Erieye_radar.

[7] Java Agent Development Framework. WWW, Accessed March 9, 2009. URL http://jade.tilab.com.

[8] Pirates take over oil tanker with british crew on board. WWW, Accessed March 9, 2009. URL http://www.guardian.co.uk/world/2008/nov/17/oil-tanker-pirates.

[9] TRACAB - Digital content for a digital world. WWW, Accessed March 9, 2009. URL http://www.tracab.com/.

[10] Amrudin Agovic, Arindam Banerjee, Auroop Ganguly, and Vladimir Protopopescu. Anomaly detection in transportation corridors using manifold embedding. In *Proceedings of the First International Workshop on Knowledge Discovery from Sensor Data*, 2007.

[11] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 1993.

[12] Sten F. Andler, Mikael Fredin, Per M. Gustavsson, Joeri van Laere, Maria Nilsson, and Pontus Svenson. SMARTracIn - A concept for spoof resistant tracking of vessels and detection of adverse intentions. In *Proceedings of the SPIE Symposium on Defense, Security and Sensing*, volume Volume 7305, 13–17 April 2009.

[13] Stefan Arnborg. Robust bayesianism: Imprecise and paradoxical reasoning. In *Proceedings of the 7th International Conference on Information Fusion*, Stockholm, Sweden, 2004.

[14] Stefan Arnborg. Robust bayesianism: Relation to evidence theory. *Journal of Advances in Information Fusion*, 1(1):63–74, 2006.

[15] Andreas Arning, Rakesh Agrawal, and Prabhakar Raghavan. A linear method for deviation detection in large databases. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 164–169. AAAI Press, 1996.

[16] Ron Azuma, Mike Daily, and Chris Furmanski. A review of time critical decision making models and human cognitive processes. In *Proceedings of the IEEE Aerospace Conference*, page 9, 4–11 March 2006.

[17] Daniel Barbará, Ningning Wu, and Sushil Jajodia. Detecting novel network intrusions using bayes estimators. In *Proceedings of the First SIAM Conference on Data Mining*, April 2001.

[18] M. Bernardo, José and Adrian F. M. Smith. *Bayesian Theory*. John Wiley and Sons, 2000.

[19] Mikael Berndtsson, Björn Lundell, Björn Olsson, and Hansson Jörgen. *Planning and Implementing Your Research Project with Success!: A Guide for Students in Computer Science and Information Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002. ISBN 1852333324.

[20] Alfs Berztiss. *Handbook of Software Engineering and Knowledge Engineering*, volume 2. World Scientific Pub. Co., 2002.

[21] Erik P. Blasch and Susan Plano. JDL level 5 fusion model: user refinement issues and applications in group tracking. In *Signal Processing, Sensor Fusion, and Target Recognition XI*, volume 4729, pages 270–279, Orlando, FL, USA, 2002. SPIE.

[22] N.A. Bomberger, B.J. Rhodes, M. Seibert, and A.M. Waxman. Associative learning of vessel motion patterns for maritime situation awareness. In *Proceedings of the 9th International Conference on Information Fusion*, page 8, Florence, Italy, July 2006. ISIF.

[23] Shyam Boriah, Varun Chandola, and Vipin Kumar. Similarity measures for categorical data: A comparative evaluation. In *Proceedings of the eighth SIAM International Conference on Data Mining*, pages 243–254, 2008.

[24] John R. Boyd. The essence of winning and losing. WWW, Accessed: March 1, 2008, 1996. URL http://www.d-ni.net/richards/boyds_ooda_loop.ppt.

[25] Paul S. Bradley, Usama Fayyad, and Cory Reina. Scaling clustering algorithms to large databases. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 9–15. AAAI Press, August 1998.

[26] Christoffer Brax and Lars Niklasson. Enhanced situational awareness in the maritime domain: An agentbased approach for situation management. In *Proceedings of the SPIE Symposium on Defense, Security and Sensing*, volume Volume 7352, Orlando, Florida, USA, 13–17 April 2009. SPIE.

[27] Christoffer Brax and Lars Niklasson. An approach for increased supply chain security by using automatic detection of anomalous vehicle behaviour. In *Proceedings of the 6th International Conference on Modelling Decisions for Artificial Intelligence*, pages 165–176, Awaji Island, Japan, November 30–December 2 2009.

[28] Christoffer Brax, Rikard Laxhammar, and Lars Niklasson. Approaches for detecting behavioural anomalies in public areas using video surveillance data. In *Proceedings of the SPIE Europe Security + Defence*, volume Volume 7113, Cardiff, Wales, United Kingdom, 15–18 September 2008. SPIE.

[29] Christoffer Brax, Lars Niklasson, and Martin Smedberg. Finding behavioural anomalies in public areas using video surveillance data. In *Proceedings of the 11th International Conference on Information Fusion*, pages 1655–1662, Cologne, Germany, June 30–July 3 2008. ISIF-IEEE.

[30] Christoffer Brax, Lars Niklasson, and Rikard Laxhammar. An ensemble approach for increased anomaly detection performance in video surveillance data. In *The 12th International Conference on Information Fusion*, pages 694–701, Seattle, WA, USA, July 6–9, 2009 2009. ISIF.

[31] Christoffer Brax, Alexander Karlsson, Sten F. Andler, Ronnie Johansson, and Lars Niklasson. Evaluating precise and imprecise state-based anomaly detectors for maritime surveillance. In *Proceedings of the 13th International Conference on Information Fusion*. ISIF-IEEE, 2010.

[32] Christoffer Brax, Alexander Karlsson, and Lars Niklasson. An empirical study of anomaly detection methods for increased situation awareness in the maritime domain. *Submitted to the Journal of Advances in Information Fusion*, 2011.

[33] Michael L. Brodie. Computer science 2.0: A new world of data management. In Christoph Koch, Johannes Gehrke, Minos N. Garofalakis, Divesh Srivastava, Karl Aberer, Anand Deshpande, Daniela Florescu, Chee Yong Chan, Venkatesh Ganti, Carl-Christian Kanne, Wolfgang Klas, and Erich J. Neuhold, editors, *Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 1161–1161, Vienna, Austria, 2007. ACM.

[34] David J. Bryant. Modernizing our cognitive model. In *Proceedings of the Command and Control Research and Technology Symposium*, San Diego, CA, 15–17 June 2004. DODCCRP.

[35] John Buford, Gabriel Jakobson, and Lundy Lewis. Extending BDI Multi-Agent Systems with Situation Management. In *Proceedings to the 9th International Conference on Information Fusion*, Florence, Italy, 10–13 July 2006. ISIF.

[36] Hung H. Bui, Dinh Q. Phung, and Svetha Venkatesh. Hierarchical hidden Markov models with general state hierarchy. In D. L. McGuinness and G. Ferguson, editors, *Proceedings of the 19th National Conference on Artificial Intelligence*, pages 324–329, San Jose, California, USA, 2004. AAAI Press / The MIT Press.

[37] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining Knowledge Discovery*, 2(2):121–167, 1998. ISSN 1384-5810.

[38] Bradley I. Buswell. High-priority technology needs. Technical report, U.S. Department of Homeland Security - Science and Technology, May 2009.

[39] Swedish Emergency Management Agency. Security Call. Ref: 1831/2008, 2008.

[40] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, 2009. ISSN 0360-0300.

[41] Paul R. Cohen. *Empirical methods for artificial intelligence*. MIT Press, Cambridge, MA, USA, 1995. ISBN 0-262-03225-2.

[42] William W. Cohen. Fast effective rule induction. In Armand Prieditis and Stuart Russell, editors, *Proceedings of the 12th International Conference on Machine Learning*, pages 115–123, Tahoe City, CA, July 9–12 1995. Morgan Kaufmann. ISBN 1-55860-377-8.

[43] Kenneth C. Cox, Stephen G. Eick, Graham J. Wills, and Ronald J. Brachman. Visual data mining: Recognizing telephone calling fraud. *Data Mining Knowledge Discovery*, 1(2):225–231, 1997.

[44] Kaustav Das and Jeff Schneider. Detecting anomalous records in categorical datasets. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 220–229. ACM Press, 2007.

[45] Belur V. Dasarathy. Information fusion - what, where, why, when, and how? editorial. *Information Fusion*, 2(2):75–76, 2001.

[46] Hannah Dee and Sergio Velastin. How close are we to solving the problem of automated visual surveillance? *Machine Vision and Applications*, 19:329–343, 2008. ISSN 0932-8092.

[47] Arthur. P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977. ISSN 00359246.

[48] Anthony Robert Dick and Michael John Brooks. Issues in automated visual surveillance. CSIRO Publishing, 2003.

[49] Thi V. Duong, Hung H. Bui, Dinh Q. Phung, and Svetha Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-Markov model. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 838–845. IEEE, 20–25 June 2005. ISBN 0-7695-2372-2.

[50] Haimonti Dutta, Chris Giannella, Kirk D. Borne, and Hillol Kargupta. Distributed Top-K Outlier Detection from Astronomy Catalogs using the DEMAC System. In *Proceedings of the 7th SIAM International Conference on Data Mining*, Minneapolis, Minnesota, USA, April 26–28 2007. SIAM.

[51] Johan Edlund, Magnus Gronkvist, Andreas Lingvall, and Egils Sviestins. Rule-based situation assessment for sea surveillance. In *Proceedings of the SPIE - Multisensor, Multisource Information Fusion: Architectures,*

Algorithms, and Applications, volume 6242, Orlando, FL, USA, 2006. SPIE.

[52] Jan Ekman and Anders Holst. Incremental stream clustering and anomaly detection. Technical report, SICS Technical Report T2008:1, 2008.

[53] Mica R. Endsley. Design and evaluation for situation awareness enhancement. In *Proceedings to the Human Factors Society 32nd Annual Meeting*, pages 97–101, Santa Monica, CA, 1988. Human Factors Society.

[54] Torkild Eriksen, Gudrun Hoye, Bjorn Narheim, and Bente Jenslokken Meland. Maritime traffic monitoring using a space-based AIS receiver. *Acta Astronautica*, 58(10):537–549, 2006.

[55] Levent Ertoz, Michael Steinbach, and Vipin Kumar. A new shared nearest neighbor clustering algorithm and its applications. In *Proceedings to the Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining*. SIAM, 2002.

[56] ESRAB. Meeting the challenge: The european security reserach agenda. Technical report, European Union, 2006.

[57] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, 1996.

[58] Tom Fawcett and Foster Provost. Activity monitoring: Noticing interesting changes in behavior. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 53–62. ACM Press, 1999.

[59] F. Fooladvandi, C. Brax, P. Gustavsson, and M. Fredin. Signature-based activity detection based on Bayesian networks acquired from expert knowledge. In *Preceedings of the 12th International Conference on Information Fusion*, pages 436–443. ISIF-IEEE, July 2009.

[60] Auroop R. Ganguly, Tailen Hsing, Rick Katz, David J. Erickson III, George Ostrouchov, Thomas J. Wilbanks, and Noel Cressie. Multivariate dependence among extremes, abrupt change and anomalies in space and time for climate applications. In *Proceedings of the International Workshop on Data Mining Methods for Anomaly Detection*, pages 25–26, Chicago, Illinois, USA, August 21 2005.

[61] D. Garagic, M. Zandipour, F. Stolle, M. Antone, and B.J. Rhodes. Hybrid neuro-bayesian spatial contextual reasoning for scene content understanding. In *Proceedings of the 12th International Conference on Information Fusion*, pages 984–989. ISIF-IEEE, July 2009.

[62] Johannes Gehrke, Raghu Ramakrishnan, and Venkatesh Ganti. RainForest - A Framework for Fast Decision Tree Construction of Large Datasets. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 416–427. Morgan Kaufmann, 1998.

[63] K. Baah George, Gray Alexander, and Harrold Mary Jean. On-line anomaly detection of deployed software: a statistical machine learning approach. ACM, 2006. pages 70–77.

[64] Vladimir Gorodetsky, Oleg Karsaev, Igor Kotenko, and Vladimir Samoilov. Multi-agent information fusion: Methodology, architecture and software tool for learning of object and situation assessment. In Johan Schubert, editor, *Proceedings of the 7th International Conference on Information Fusion*, volume I, pages 346–353, Stockholm, Sweden, June 2004. ISIF. ISBN 91-7056-115-X.

[65] Peter Gärdenfors. *Conceptual Spaces - The Geometry of Thought*. The MIT Press, 2000.

[66] Marco Guerriero, Peter Willett, Stefano Coraluppi, and Craig Carthel. Radar/AIS Data Fusion and SAR tasking for Maritime Surveillance. In *Proceedings of the 11th International Conference on Information Fusion*, pages 1650–1654, Cologne, Germany, June 30–July 3 2008. ISIF-IEEE.

[67] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. In *Proceedings of the 15th International Conference on Data Engineering*, pages 512–521. IEEE, 2000.

[68] Daniel Gutchess, Neal Checka, and Magnus S. Snorrason. Learning patterns of human activity for anomaly detection. In *Proceedings of the SPIE Intelligent Computing: Theory and Applications V*, volume 6560, Orlando, FL, USA, 2007. SPIE.

[69] David L. Hall and Sonya A.H. McMullen. *Mathematical Techniques in Multisensor Data Fusion*. Artech House Inc., Boston, second edition, 2004. ISBN 1-58053-335-3.

[70] David L. Hall, Michael Mcneese, James Lunas, and Tracy Mullen. A framework for dynamic hard/soft fusion. In *Proceedings of the 11th International Conference on Information Fusion*, pages 85–92, Cologne, Germany, June 30–July 3 2008. ISIF-IEEE.

[71] Mary Jane M. Hall, Sonya A. Hall, and Timothy Tate. Removing the HCI Bottleneck: How the Human-Computer Interface (HCI) Affects the Performance of Data Fusion Systems. In *Proceedings to the MSS National Symposium on Sensor and Data Fusion*, pages 89–104, San Diego, CA, 2000.

[72] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster based local outliers. *Pattern Recognition Letters*, 2003:9–10, 2003.

[73] D. Heckerman. A tutorial on learning with bayesian networks, 1995. URL citeseer.ist.psu.edu/heckerman96tutorial.html.

[74] *Recommendation ITU-R M.1371-4. Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile band.* International Telecommunication Union, 2010.

[75] Gabriel Jakobson, Lundy Lewis, Christopher J. Matheus, Mieczyslaw M. Kokar, and John Buford. Overview of situation management at SIMA 2005. In *Proceedings of the IEEE Military Communications Conference*, volume 3, pages 1630–1636, Atlantic City, NJ, USA, 2005. IEEE. ISBN 0-7803-9393-7.

[76] Gabriel Jakobson, John Buford, and Lundy Lewis. Situation management: Basic concepts and approaches. In Vasily V. Popovich, Manfred Schrenk, and Kyrill V. Korolenko, editors, *Information Fusion and Geographic Information Systems*, pages 18–33. Berlin, Heidelberg, 2007. ISBN 9783540376293.

[77] F. Johansson and G. Falkman. Detection of vessel anomlies - a bayesian network approach. In *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2007.

[78] I.T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[79] John E. Gaffney Jr and Jacob W. Ulvila. Evaluation of intrusion detectors: A decision theory approach. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 50–61. IEEE, May 2001.

[80] Hillol Kargupta, Weiyun Huang, Krishnamoorthy Sivakumar, and Erik Johnson. Distributed clustering using collective principal component analysis. *Knowledge and Information Systems*, 3:2001, 1999.

[81] Alexander Karlsson, Ronnie Johansson, and Sten F. Andler. On the behavior of the robust bayesian combination operator and the significance of discounting. In *Proceedings of the 6th International Symposium on*

Imprecise Probability: Theories and Applications, Durham, U.K., 2009. ISIPTA.

[82] Eamonn Keogh, Stefano Lonardi, and Chotirat Ann Ratanamahatana. Towards parameter-free data mining. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 206–215. ACM Press, 2004.

[83] Andreas Kind, Marc Ph. Stoecklin, and Xenofontas Dimitropoulos. Histogram-based traffic anomaly detection. *IEEE Transactions on Network and Service Management*, 6(2):110–121, 2009.

[84] Teuvo Kohonen. *Self-organizing maps*. Springer, 3rd edition, 2000.

[85] James B. Kraiman, Scott L. Arouh, and Michael L. Webb. Automated anomaly detection processor. In *Enabling Technologies for Simulation Science VI*, volume 4716, pages 128–137, Orlando, FL, USA, 2002. SPIE.

[86] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley and Sons Inc., 2004. ISBN 978-0-471-21078-8.

[87] Rikard Laxhammar. Anomaly detection for sea surveillance. In *Proceedings of the 11th International Conference on Information Fusion*, pages 55–62, Cologne, Germany, June 30–July 3 2008. ISIF-IEEE.

[88] Rikard Laxhammar, Göran Falkman, and Egils Sviestins. Anomaly detection in sea traffic - a comparison of the Gaussian Mixture Model and the Kernel Density Estimator. In *Proceedings of the 12th International Conference on Information Fusion*, pages 756–763, Seattle, WA, USA, July 6–9 2009. ISIF-IEEE.

[89] Aleksandar Lazarevic, Levent Ertoz, Vipin Kumar, Aysel Ozgur, and Jaideep Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the 3rd SIAM international conference on data mining.*, pages 25–36, San Francisco, CA, USA, 2003. SIAM.

[90] Kingsly Leung and Christopher Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the 28th Australasian CS Conference*, volume 38, pages 333–342. CRPITV, 2005.

[91] Yair Levy and Timothy J. Ellis. A systems approach to conduct an effective literature review in support of information systems research. *Informing Science Journal*, 9:181–212, 2006.

[92] Xiaolei Li and Jiawei Han. Mining Approximate Top-K Subspace Anomalies in Multi-Dimensional Time-Series Data. In Christoph Koch, Johannes Gehrke, Minos N. Garofalakis, Divesh Srivastava, Karl Aberer, Anand Deshpande, Daniela Florescu, Chee Yong Chan, Venkatesh Ganti, Carl-Christian Kanne, Wolfgang Klas, and Erich J. Neuhold, editors, *Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 447–458. ACM, 2007.

[93] Martin E. Liggins, David L. Hall, and James Llinas. *Handbook of Multisensor Data Fusion: Theory and Practice*. CRC Press, 1 edition, 2009. ISBN 0849323797. URL http://www.worldcat.org/isbn/0849323797.

[94] Huan Liu, Farhad Hussain, Chew L. Tan, and Manoranjan Dash. Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–423, October 2002.

[95] James Llinas, Christopher L. Bowman, Galina L. Rogova, Alan N. Steinberg, Edward L. Waltz, and Frank E. White. Revisions and extensions to the JDL data fusion model II. In Johan Schubert, editor, *Proceedings of the 7th International Conference on Information Fusion*, volume II, pages 1218–1230, Stockholm, Sweden, June 2004. ISIF. ISBN 91-7056-116-8.

[96] Carl G. Looney. Exploring fusion architecture for a common operational picture. *Information Fusion*, 2(4):251–260, 2001.

[97] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart. The mahalanobis distance. *Chemometrics and Intelligent Laboratory Systems*, 50 (1):1–18, 2000. ISSN 0169-7439.

[98] Ofer Matan. On voting enssembles of classifiers (extended abstract). In *Proceedings of AAAI-96 workshop on Integrating Multiple Learned Models*, pages 84–88. AAAI, 1996.

[99] Lars Niklasson, Maria Riveiro, Fredrik Johansson, Anders Dahlbom, Göran Falkman, Tom Ziemke, Christoffer Brax, Thomas Kronhamn, Martin Smedberg, Håkan Warston, and Per M. Gustavsson. A unified situation analysis model for human and machine situation awareness. In *Proceedings of the 11th International Conference on Information Fusion*, pages 454–461, Cologne, Germany, June 30–July 3 2008. ISIF-IEEE.

[100] Thaper Nitin, Guha Sudipto, Indyk Piotr, and Koudas Nick. Dynamic multidimensional histograms. ACM, 2002. 564741 428-439.

[101] Camilla Nyqvist and Patrik Bergsten. šSecure And Efficient Intermodal Transports - Pilot Project In The Port Of Gothenburg, Sweden. In *Proceedings of the 15th World Congress on Intelligent Transport Systems*, New York, USA, November 16–20 2008. Omnipress.

[102] Lucas Parra, Gustavo Deco, and Stefan Miesbach. Statistical independence and novelty detection with information preserving nonlinear maps. *Neural Computation*, 8:260–269, 1995.

[103] Animesh Patcha and Jung-Min Park. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12):3448–3470, August 2007. ISSN 1389-1286.

[104] Edwin P. Pednault. Representation is everything. *Communcatins of the ACM*, 43(8):80–83, 2000. ISSN 0001-0782.

[105] Leonid Portnoy, Eleazar Eskin, and Salvatore J. Stolfo. Intrusion detection with unlabeled data using clustering. In *Proceedings to the ACM Workshop on Data Mining Applied to Security*, USA, 2001. ACM Press.

[106] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In Alex Waibel and Kai-Fu Lee, editors, *Readings in speech recognition*, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990. ISBN 1-55860-124-4.

[107] Thomas E. Reeves, Orion J. Welch, and Sandra Welch. 10-fold cross-validation for discriminant analysis. In *Proceedings of the Decision Science Institute 1999 Annual Meeting*, November 1999.

[108] B.J. Rhodes, N.A. Bomberger, M. Seibert, and A.M. Waxman. Maritime situation monitoring and awareness using learning mechanisms. In *Proceedings of the IEEE Military Communications Conference*, volume 1, pages 646–652, Atlantic City, New Jersey, 17–20 October 2005. IEEE. ISBN 0-7803-9393-7.

[109] B.J. Rhodes, N.A. Bomberger, and M. Zandipour. Probabilistic associative learning of vessel motion patterns at multiple spatial scales for maritime situation awareness. In *Proceedings of the 10th International Conference on Information Fusion*, pages 1–8. ISIF, July 2007.

[110] Bradley J. Rhodes, Neil A. Bomberger, Majid Zandipour, Lauren H. Stolzar, Denis Garagic, James R. Dankert, and Michael Seibert. Anomaly detection & behavior prediction: Higher-level fusion based on computational neuroscientific principles. In Nada Milisavljevic, editor, *Sensor and Data Fusion*, pages 323–336. InTech, 2009.

[111] B. Ristic, B. La Scala, M. Morelande, and N. Gordon. Statistical analysis of motion patterns in AIS Data: Anomaly detection and motion prediction. In *Proceedings to the 11th International Conference on Information Fusion*, pages 40–46. ISIF, June 30–July 3 2008.

[112] Maria Riveiro, Göran Falkman, and Tom Ziemke. Improving maritime anomaly detection and situation awareness through interactive visualization. In *Proceedings of the 11th International Conference on Information Fusion*, pages 47–54, Cologne, Germany, June 30–July 3 2008. ISIF-IEEE.

[113] Joseph L. Rodgers and Alan W. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[114] Jean Roy. From data fusion to situation analysis. In *Proceedings of the 4th International Conference on Information Fusion*, Montreal, Canada, 2001. ISIF.

[115] Jean Roy. Anomaly detection in the maritime domain. In *Procedings of the SPIE*, volume 6945. SPIE, 2008.

[116] Jean Roy and Michal Davenport. Categorization of maritime anomalies for notification and alerting purpose. In *Proceedings of the NATO Workshop on Data Fusion and Anomaly Detection for Maritime Situation Awareness*, La Spezia, Italy, 2009. NURC.

[117] Gunnar Rätsch, Sebastian Mika, Bernhard Schölkopf, and Klaus-Robert Müller. Constructing Boosting Algorithms from SVMs: an Application to One-Class Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1184–1199, 2002.

[118] Dymitr Ruta and Bogdan Gabrys. An overview of classifier fusion methods. *Computing and Information Systems*, 7(1):1–10, 2000.

[119] J. Salerno, M. Hinman, and D. Boulware. Building a framework for situation awareness. In *Proceedings to the 7th International Conference on Information Fusion*, pages 182–193, Stockholm, Sweden, June 2004. ISIF.

[120] Hanan Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, 2006.

[121] Michael Seibert, Bradley J. Rhodes, Neil A. Bomberger, Patricia O. Beane, Jason J. Sroka, Wendy Kogel, William Kreamer, Chris Stauffer, Linda Kirschner, Edmond Chalom, Michael Bosse, and Robert Tillson. SeeCoast port surveillance. In Michael J. DeWeert, Theodore T. Saito, and Harry L. Guthmuller, editors, *Proceedings of the SPIE - Photonics*

for Port and Harbor Security II, volume 6204, page 62040B. SPIE, June 2006.

[122] M. Setnes, R. Babuska, and H.B. Verbruggen. Rule-based modeling: precision and transparency. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 28(1):165–169, February 1998. ISSN 1094-6977.

[123] Xiuyao Song, Mingxi Wu, Christopher Jermaine, and Sanjay Ranka. Conditional anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 19(5):631–645, 2007. ISSN 1041-4347.

[124] Claudio De Stefano, Carlo Sansone, and Mario Vento. To Reject or Not to Reject: That is the Question - An Answer in Case of Neural Classifiers. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 30(1):84–94, 2000.

[125] Alan N. Steinberg. Context-sensitive data fusion using structural equation modeling. In *Proceedings of the 12th International Conference on Information Fusion*, pages 725 –731. ISIF, July 2009.

[126] Alan. N. Steinberg, Christopher L. Bowman, and Franklin E. White. Revisions to the JDL data fusion model. In *Proceedings of SPIE AeroSense (Sensor Fusion: Architectures, Algorithms and Applications III)*, pages 430–441. SPIE, 1999.

[127] George W. Stimson. *Introduction to Airborne Radar*. SciTech Publishing, 2 edition, 1998.

[128] Salvatore J. Stolfo, Wei Fan, and Wenke Lee. Cost-Based Modeling for Fraud and Intrusion Detection Results from the JAM Project. In *Proceedings of the DARPA Information Survivability Conference*. DARPA, 2000.

[129] Hangal Sudheendra and S. Lam Monica. Tracking down software bugs using automatic anomaly detection. In *Proceedings of the 24th International Conference on Software Engineering*, pages 291–301. ACM Press, 2002. 581377 291-301.

[130] Jimeng Sun, Yinglian Xie, Hui Zhang, and Christos Faloutsos. Less is more: Compact matrix decomposition for large sparse graphs. In *Proceedings of the SIAM International Conference on Data Mining*. SIAM, 2007.

[131] Per Svensson. Technical survey and forecast for information fusion. In *Proceedings of the RTO IST Symposium on Military Data and Information Fusion*, Prague, Czech Republic, 20–22 October 2003.

[132] Pang-Ning Tan, Mickael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison-Wesley, Boston, 2006. ISBN 0-321-32136-7.

[133] Sarah M. Taylor. How much information is enough? In *Proceedings to the 10th International Command and Control Research and Technology Symposium*, San Diego, CA, USA, 2005. DODCCRP.

[134] Joeri van Laere and Maria Nilsson. Evaluation of a workshop to capture knowledge from subject matter experts in maritime surveillance. In *Proceedings to the 12th International Conference on Information Fusion*, pages 171–178. ISIF, 2009.

[135] Hodge Victoria and Austin Jim. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004. ISSN 0269-2821.

[136] Jeffrey Scott Vitter. An efficient algorithm for sequential random sampling. *ACM Trans. Math. Softw.*, 13:58–67, March 1987. ISSN 0098-3500.

[137] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann Publishers, San Francisco, CA, 2nd edition, 2005. ISBN 0-12-088407-0.

Publications *in the series*
Örebro Studies in Technology

1. Bergsten, Pontus (2001) *Observers and Controllers for Takagi – Sugeno Fuzzy Systems*. Doctoral Dissertation.
2. Iliev, Boyko (2002) *Minimum-time Sliding Mode Control of Robot Manipulators*. Licentiate Thesis.
3. Spännar, Jan (2002) *Grey box modelling for temperature estimation*. Licentiate Thesis.
4. Persson, Martin (2002) *A simulation environment for visual servoing*. Licentiate Thesis.
5. Boustedt, Katarina (2002) *Flip Chip for High Volume and Low Cost – Materials and Production Technology*. Licentiate Thesis.
6. Biel, Lena (2002) *Modeling of Perceptual Systems – A Sensor Fusion Model with Active Perception*. Licentiate Thesis.
7. Otterskog, Magnus (2002) *Produktionstest av mobiltelefonantenner i mod-växlande kammare*. Licentiate Thesis.
8. Tolt, Gustav (2003) *Fuzzy-Similarity-Based Low-level Image Processing*. Licentiate Thesis.
9. Loutfi, Amy (2003) *Communicating Perceptions: Grounding Symbols to Artificial Olfactory Signals*. Licentiate Thesis.
10. Iliev, Boyko (2004) *Minimum-time Sliding Mode Control of Robot Manipulators*. Doctoral Dissertation.
11. Pettersson, Ola (2004) *Model-Free Execution Monitoring in Behavior-Based Mobile Robotics*. Doctoral Dissertation.
12. Överstam, Henrik (2004) *The Interdependence of Plastic Behaviour and Final Properties of Steel Wire, Analysed by the Finite Element Metod*. Doctoral Dissertation.
13. Jennergren, Lars (2004) *Flexible Assembly of Ready-to-eat Meals*. Licentiate Thesis.
14. Jun, Li (2004) *Towards Online Learning of Reactive Behaviors in Mobile Robotics*. Licentiate Thesis.
15. Lindquist, Malin (2004) *Electronic Tongue for Water Quality Assessment*. Licentiate Thesis.
16. Wasik, Zbigniew (2005) *A Behavior-Based Control System for Mobile Manipulation*. Doctoral Dissertation.

17. Berntsson, Tomas (2005) *Replacement of Lead Baths with Environment Friendly Alternative Heat Treatment Processes in Steel Wire Production*. Licentiate Thesis.

18. Tolt, Gustav (2005) *Fuzzy Similarity-based Image Processing*. Doctoral Dissertation.

19. Munkevik, Per (2005) *"Artificial sensory evaluation – appearance-based analysis of ready meals"*. Licentiate Thesis.

20. Buschka, Pär (2005) *An Investigation of Hybrid Maps for Mobile Robots*. Doctoral Dissertation.

21. Loutfi, Amy (2006) *Odour Recognition using Electronic Noses in Robotic and Intelligent Systems*. Doctoral Dissertation.

22. Gillström, Peter (2006) *Alternatives to Pickling; Preparation of Carbon and Low Alloyed Steel Wire Rod*. Doctoral Dissertation.

23. Li, Jun (2006) *Learning Reactive Behaviors with Constructive Neural Networks in Mobile Robotics*. Doctoral Dissertation.

24. Otterskog, Magnus (2006) *Propagation Environment Modeling Using Scattered Field Chamber*. Doctoral Dissertation.

25. Lindquist, Malin (2007) *Electronic Tongue for Water Quality Assessment*. Doctoral Dissertation.

26. Cielniak, Grzegorz (2007) *People Tracking by Mobile Robots using Thermal and Colour Vision*. Doctoral Dissertation.

27. Boustedt, Katarina (2007) *Flip Chip for High Frequency Applications – Materials Aspects*. Doctoral Dissertation.

28. Soron, Mikael (2007) *Robot System for Flexible 3D Friction Stir Welding*. Doctoral Dissertation.

29. Larsson, Sören (2008) *An industrial robot as carrier of a laser profile scanner. – Motion control, data capturing and path planning*. Doctoral Dissertation.

30. Persson, Martin (2008) *Semantic Mapping Using Virtual Sensors and Fusion of Aerial Images with Sensor Data from a Ground Vehicle*. Doctoral Dissertation.

31. Andreasson, Henrik (2008) *Local Visual Feature based Localisation and Mapping by Mobile Robots*. Doctoral Dissertation.

32. Bouguerra, Abdelbaki (2008) *Robust Execution of Robot Task-Plans: A Knowledge-based Approach*. Doctoral Dissertation.

33. Lundh, Robert (2009) *Robots that Help Each Other: Self-Configuration of Distributed Robot Systems*. Doctoral Dissertation.

34. Skoglund, Alexander (2009) *Programming by Demonstration of Robot Manipulators*. Doctoral Dissertation.

35. Ranjbar, Parivash (2009) *Sensing the Environment: Development of Monitoring Aids for Persons with Profound Deafness or Deafblindness*. Doctoral Dissertation.

36. Magnusson, Martin (2009) *The Three-Dimensional Normal-Distributions Transform – an Efficient Representation for Registration, Surface Analysis, and Loop Detection*. Doctoral Dissertation.

37. Rahayem, Mohamed (2010) *Segmentation and fitting for Geometric Reverse Engineering. Processing data captured by a laser profile scanner mounted on an industrial robot*. Doctoral Dissertation.

38. Karlsson, Alexander (2010) *Evaluating Credal Set Theory as a Belief Framework in High-Level Information Fusion for Automated Decision-Making*. Doctoral Dissertation.

39. LeBlanc, Kevin (2010) *Cooperative Anchoring – Sharing Information About Objects in Multi-Robot Systems*. Doctoral Dissertation.

40. Johansson, Fredrik (2010) *Evaluating the Performance of TEWA Systems*. Doctoral Dissertation.

41. Trincavelli, Marco (2010) *Gas Discrimination for Mobile Robots*. Doctoral Dissertation.

42. Cirillo, Marcello (2010) *Planning in Inhabited Environments: Human-Aware Task Planning and Activity Recognition*. Doctoral Dissertation.

43. Nilsson, Maria (2010) *Capturing Semi-Automated Decision Making: The Methodology of CASADEMA*. Doctoral Dissertation.

44. Dahlbom, Anders (2011) *Petri nets for Situation Recognition*. Doctoral Dissertation.

45. Ahmed, Muhammad Rehan (2011) *Compliance Control of Robot Manipulator for Safe Physical Human Robot Interaction*. Doctoral Dissertation.

46. Riveiro, Maria (2011) *Visual Analytics for Maritime Anomaly Detection*. Doctoral Dissertation.

47. Rashid, Md. Jayedur (2011) *Extending a Networked Robot System to Include Humans, Tiny Devices, and Everyday Objects*. Doctoral Dissertation.

48. Zain-ul-Abdin (2011) *Programming of Coarse-Grained Reconfigurable Architectures*. Doctoral Dissertation.

49. Wang, Yan (2011) *A Domain-Specific Language for Protocol Stack Implementation in Embedded Systems*. Doctoral Dissertation.

50. Brax, Christoffer (2011) *Anomaly Detection in the Surveillance Domain*. Doctoral Dissertation.