University Of Genova

Queen Mary University of London

Joint Doctoral degree in Interactive and Cognitive Environments

---

Ego things: Networks Of Self-Aware Intelligent Objects

Multi-modal dynamic Bayesian models for self-awareness and collective awareness in agents' network

---

by
Divya Thekke Kanapram

A thesis submitted for the degree of Doctor of Philosophy

June 2021

Abstract

There is an increasing demand for developing intelligence and awareness in artificial agents in recent days to improve autonomy, robustness, and scalability, and it has been investigated in various research fields such as machine learning, robotics, software engineering, etc. Moreover, it is crucial to model such an agent's interaction with the surrounding environment and other agents to represent collaborative tasks. In this thesis, we have proposed several approaches to developing multi-modal self-awareness in agents and multi-modal collective awareness (CA) for multiple networked intelligent *agents* by focusing on the functionality to detect abnormal situations.

The first part of the thesis is proposed a novel approach to build self-awareness in dynamic agents to detect abnormalities based on multi-sensory data and feature selection. By considering several sensory data features, learned multiple inference models and facilitated obtaining the most distinct features for predicting future instances and detecting possible abnormalities. The proposed method can select the optimal set features to be shared in networking operations such that state prediction, decision-making, and abnormality detection processes are favored.

In the second part, proposed different approaches for developing collective awareness in an agents network. Each agent of a network is considered an Internet of Things (IoT) node equipped with machine learning capabilities. The collective awareness aims to provide the network with updated causal knowledge of the state of execution of actions of each node performing a joint task, with particular attention to anomalies that can arise. Data-driven dynamic Bayesian models learned from multi-sensory data recorded during the normal realization of a joint task (agent network experience) are used for distributed state estimation of *agents* and detection of abnormalities. Moreover, the effects of networking protocols and communications in the estimation of state and abnormalities are analyzed.

Finally, the abnormality estimation is performed at the model's different abstraction levels and explained the models' interpretability. In this work, interpretability is the capability to use anomaly data to modify the model to make inferences accurately in the future.

Acknowledgements

First of all, I want to dedicate this work to my mother, Devaki Karat Analakkat, and my father, who never saw this adventure. I would not be able to achieve this without my mother's continuous motivation and support; she was always with me in all the ups and downs of my research and life with a lot of patience and encouragement. My father always stressed the importance of education, and the never-ending conversations I had with him shaped my values and made me the person I am today. My grandmother, Aditi, has been a source of constant and unconditional love for as long as I can remember, and I am extremely grateful to her, who taught me about what it means to be a good person.

I am incredibly grateful to my supervisors, Prof. Carlo S. Regazzoni, Prof. Mario Marchese, and Dr. Eliane L Bodanese, for their invaluable advice, continuous support, and patience during my Ph.D. study. Their immense knowledge and ample experience have encouraged me in my academic research and daily life. I am deeply grateful to Prof. Carlo S. Regazzoni for his unwavering support and belief in me. He was always available for the discussions and supported me during the holidays whenever I faced difficulties in my research. I have been fortunate to have supervisors who cared so much about my work and promptly responded to my questions and queries. I would like to thank also Dr. Lucio Marcenaro for his continuous motivation and support during my Ph.D. I want to thank Dr. Raul Mondragon, my independent assessor in QMUL, for his valuable suggestions and comments in each stage of my Ph.D.

I would like to thank our collaborators in UC3M, Spain, especially Dr. David and Dr. Pablo, for their support by providing datasets for my research. I want to acknowledge my colleagues for a cherished time spent together in the lab and social settings. In particular, Fabio Patrone, Damian

Campo, Muhammad Baydoun, Mahdyar Ravanbakhsh, Muhammad Farrukh, Ali Krayani, Hafza Iqbal, Sheida Nozari, and Giulia Slavic. I would love to thank my colleagues at the Queen Mary University of London, where I spent half of my Ph.D. period. I also thank my friends Pradeep, Imran, and Muthalib for their moral support, patience, and curiosity to listen to my never-ending conversations about research.

The last two people I want to thank are my brother Vishnu and sister in law Sujatha. Without their tremendous understanding and encouragement in the past few years, completing my study would be impossible.

Contents

List of Figures

List of Tables

List of abbreviations

AUC	area under the curve
ACC	accuracy
BN	Bayesian network
CA	collective awareness
CDBN	collective dynamic Bayesian network
CSMA/CA	carrier-sense multiple access with collision avoidance
DAG	directed acyclic graph
DBN	dynamic Bayesian network
DKF	distributed Kalman filter
D-MJPF	distributed Markov jump particle filter
DSRC	dedicated short range communications
GDBN	generative dynamic Bayesian network
GE	generalized error
GNG	growing neural gas
GUI	graphical user interface
HD	Hellinger distance
HMM	hidden Markov model
IMM	interacting multiple models
KF	Kalman filter
KFM	Kalman filter model
KL	Kullback-Leibler distance
KCF	Kalman-consensus filter
MJPF	Markov jump particle filter
MJLS	Markov jump linear system
MSE	mean squared error
NG	neural gas
PF	particle filter
ROC	receiver operating characteristic
SA	self-awareness
UKF	unmotivated Kalman filter
WSN	wireless sensor network

Chapter 1

Introduction

This chapter presents the motivation of this thesis, objectives of the research, the main contributions, Ph.D. publications, and the thesis structure.

1.1 The importance of self-awareness and collective awareness in agents network

The Internet of things (IoT) related technologies has advanced well beyond our imaginations in the past few years. At present, billions of physical devices worldwide are connected to the Internet, and most of them can collect and share large amounts of data. In general, any device can be thought of as an IoT device if it has networking capabilities. A usual IoT device can vary from a child's toy to a driver-less vehicle. An IoT with smart objects has many applications in the field of surveillance [30], transportation [46], crowd monitoring [45], etc.

In general, the IoT has not yet reached a desired level of maturity, and challenges are still open, such as computation constraints, heterogeneity, data storage, autonomous capabilities, security, etc. In this thesis, we refer to smart IoT objects as agents as they present different behavior characteristics and interactions with other objects and/or humans. One of the most crucial challenges is the lack of proper models representing the agent behaviors and their causal relationships to the surrounding environments and other objects [89]. Modeling dynamic agents' interaction with the surrounding environment and other agents is vital for predicting future unseen dynamics accurately to avoid unwanted situations.

The interaction model should be capable of spanning over variables at different abstraction levels to allow, for example, better explainability of autonomous agent's choices both online and offline (ex-post). Moreover, the learning of representation can perform in a data-driven way from observed sensory data when the agent performs an experience for the first time (possibly driven by an external control) [54]. With machine learning algorithms and signal processing techniques, the IoT nodes can include such learning capabilities. Artificial self-awareness (SA) has an essential role in this framework.

The term Internet of Vehicles (IoV) has been defined whenever smart physical objects are vehicles [40]. Due to the rise of the population in the cities, the number of vehicles has grown exponentially, leading to congestion and pollution. Consequently, road accidents have increased dramatically for numerous reasons such as lack of contextual data, distracted and reckless driving, adverse weather conditions, animal crossing, unsafe lane changes, etc. These factors show the importance of making the vehicles "self-aware" to ensure safety in driving. Future autonomous vehicles will use these self-awareness processes to solve different tasks in different environments that never were considered previously for decision making or problem-solving, such as trajectory planning or collision avoidance, or abnormality detection. The new learning capabilities from observing the environment or the autonomous vehicles' own states of behavior create new possibilities for problem-solving in new situations of real traffic scenarios. Therefore, developing self-aware models will improve the general decision and the navigation in autonomous vehicles facilitating the improvement of the incremental self-capabilities, such as fault-tolerant decisions based on own perception or communication capabilities in dynamic environments.

Self-awareness is a broad concept that defines the agent's ability to focus on the inner self-state in relation to the external environment (i.e., cognitive property) [84]. In the case of artificial agents like intelligent vehicles (IVs) [104], the concept of SA is an ability to observe themselves and the surrounding environment through the various exteroceptive and proprioceptive sensors and process the sensory data to learn and maintain a contextual representation of the system [84]. The physical architecture for a single self-aware autonomous agent, i.e., ego-thing, is shown in Fig. 1.1. Ego-thing can be defined as intelligent autonomous entities that can perceive their

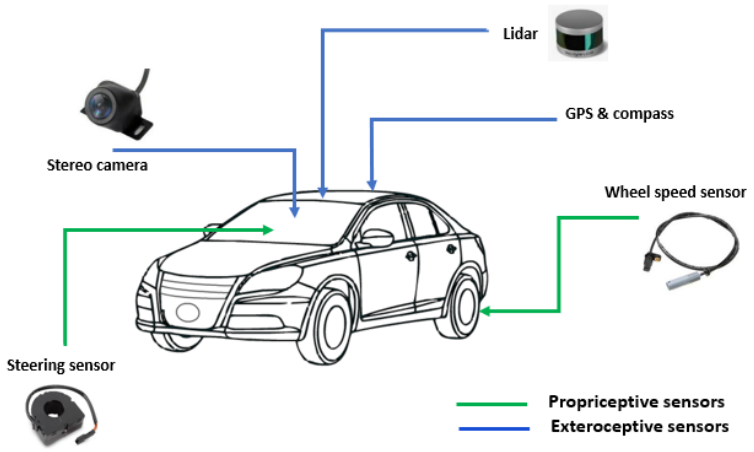


Figure 1.1: Physical architecture for a self aware agent (ego-thing.)

internal as well as external parameters and adapt themselves when they face abnormal situations [51]. In this thesis, ego-thing, agent, object, and vehicle are used as synonymous. Nowadays, machine learning provides an extensive set of methods and techniques to estimate SA models from data sequences. This work firstly considered developing self-awareness in agents and provided a methodology by which collective awareness (CA) of a group of agents could be defined and achieved. The interacting agents' network is shown in Fig. 1.2. Each dynamic agent interacts with the surrounding environment and other agents in the network. Moreover, the methodology shows how the proposed techniques can be suitable for jointly building the individual and collective representation of the state of development of a task concerning SA models learned from previous experiences.

Bayesian Network (BN) techniques are the reference approach used in this work to represent awareness models. Models are composed of multiple variables, hierarchically organized into layers. The sensors perceive the layers associated with the observed variables. In contrast, hidden layers represent variables with direct or indirect causal relationships with observations at different abstraction levels.

Dynamic BNs introduce temporal links that connect BNs variables at successive time instants, allowing to describe causal and temporal relationships, i.e., behaviors of the same object as represented by a dynamic series of

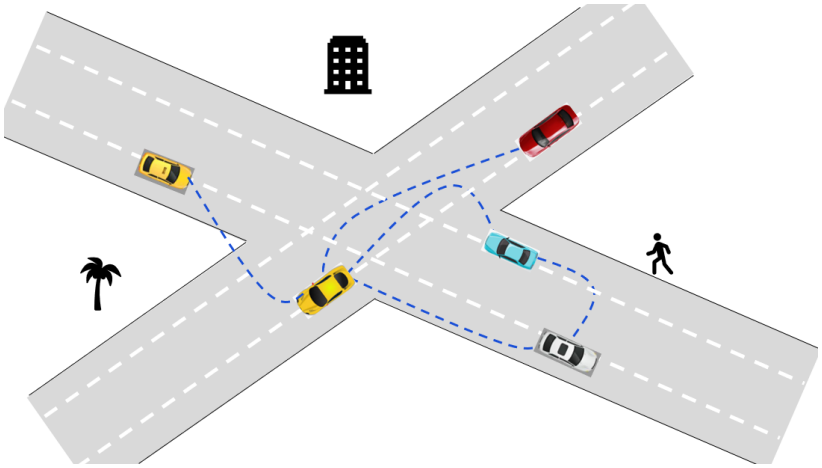


Figure 1.2: Interacting agents network (ego-things network).

probability states of variables in time. DBNs allow an agent to explain the temporal series of observed sensor data at different abstraction levels thanks to the global model's generative property. The same property can apply for simultaneous observation of objects doing collaborative tasks. Links among DBNs related to each object can explain coupling conditional probabilities describing reciprocal influences among such objects. This generative capability makes coupled DBNs attractive for composing the basis of SA representation in an agent. However, as SA models are data-driven, i.e., learnable from the data inference, DBNs' generative capabilities must be augmented by incrementally learning new DBNs from sensor observations when such observations correspond to new experiences. This implies that inference on such models also includes anomaly detection and incremental learning steps in addition to classical Bayesian prediction-estimation filtering.

Let us suppose that SA knowledge at a certain point of an agent's life is defined using DBN models learned from sensorial data connected to the agent's past experiences. The next step is to define a Bayesian inference process capable of allowing the agents to continuously check and monitor whether available DBN generative models can predict well the current observations in doing the current task. When a new experience comes that the agent cannot use the embedded knowledge of learned models to predict and reliably estimate its own context state (abnormal situations), it requires incremental learning of new models to enrich its own SA memory.

Another important area to be focused on is to assess the impact of re-

alistic information exchange among objects on the abnormality detection feature of collective awareness. With reliable and efficient communication, agents should share ground truth observations acquired in a distributed way by each of them with all other agents in the network. Each agent should appropriately dispatch the ground truth observations received from each remote transmitting agent to estimate the possible presence of abnormalities. In this way, each agent can estimate local and global abnormality conditions that can arise in any of the agents that compose the network. Self-awareness of single agents can become collective awareness of the agent's network. Different distributed communication schemes at increasing complexity can be devised to reach such a collective awareness.

1.2 Research questions

This thesis provides different approaches to building self-awareness in agents and collective awareness for an agent's network for state estimation and abnormality detection, discussed in the next chapters. The work aims to answer the following research questions:

1.2.1 Can an artificial agent be self-aware?

Self-awareness (SA) is a broad concept that describes the cognitive property of a biological agent. In artificial agents, the concept of SA is an ability to observe themselves and the surrounding environment through the various exteroceptive and proprioceptive sensors and process the sensory data to learn and maintain a contextual representation of the system [84]. Nowadays, the emergent techniques and algorithms in Machine learning allow for the learning of data-driven models that can provide self-awareness functionalities.
The first part of this thesis presents an implementation example that achieves self-awareness. The DBN models are learned from multi-sensory observed data sequences; such a model can predict the agents' future unseen dynamics and detect abnormal situations.

1.2.2 Distributed state estimation in network of agents

Distributed state estimation and tracking are fundamental collaborative information processing problems in wireless sensor networks (WSNs). Multi-sensor fusion and tracking problems have a long history in signal processing, control theory, and robotics [6, 12, 11, 36]. Moreover, the distributed state estimation issues in wireless networks with packet-loss have been the center of much attention lately [92, 47, 87]. There has been a significant development in the study of Kalman filtering in the presence of data packet drops [61, 68, 79, 87]. The recent advances in the WSN technology also boost the study of a distributed Kalman filter (DKF) [5, 3, 74, 94], where each sensor node in the WSN can compute local estimates via Kalman filtering based on its own observations and the information sent from its neighboring sensors. The existing literature shows that it lacked a proper model to represent a group of agents' behavior and underlying reasons in different situations.

This work investigates and proposes solutions for distributed state estimation in an agents' network with the help of machine learning algorithms, signal processing techniques, and dynamic Bayesian filter.

1.2.3 Can collective awareness be implemented in an agents' network?

This thesis proposes that a 'self-awareness' functionality of agents can be extended to 'collective awareness' by utilizing the exteroceptive and proprioceptive sensory data shared among the agents in the network performing co-operative tasks. This thesis proposes various approaches to learn collective awareness models that represent agents' joint tasks. A co-operative communication scheme is used in the experimental scenarios to exchange sensory data among agents.

1.3 Contribution of the thesis

Different methodologies are presented in order to achieve the learning and testing of self-awareness and collective awareness in an ego-things' network. The first part of the thesis proposes and tests a method for agents to learn self-awareness models in relation to specific exteroceptive and proprioceptive sensory data. The second part presents how agents can learn multi-modal

collective awareness models for agents' networks.

The main contributions of this thesis can be summarized as follows.

1. A method is proposed for learning multi-modal self-awareness models from low dimensional data sequences collected by intelligent agents' various exteroceptive and proprioceptive sensors. The proposed strategy recognizes the different features from a sensory data set and selects the considered task's best feature.

2. Several approaches for learning multi-modal collective awareness models from low dimensional data sequences of a network of intelligent entities are proposed and tested. For the inferences, a Markov jump particle filter (MJPF) based on generalized DBN models is used for distributed state estimation and extended to become able to detect abnormalities at different abstraction levels. The metrics considered to estimate abnormalities at different abstraction levels are Hellinger distance, innovation metric, and Kullback Leibler (KL) distance.

3. The robustness of the distributed state estimation and abnormality detection feature of the models is investigated in a network built using a realistic communication channel model. Firstly, the reliability and accuracy of the abnormality detection are measured under the hypothesis of perfect communication (i.e., no data loss and transmission delays). Secondly, the reliability and accuracy of the abnormality detection is measured under the use of different protocols and channel conditions (i.e., in the presence of packet losses and transmission delays of the communication channel among objects.

4. Finally, interpretability features of interactive machine learning models have been exploited by presenting and discussing multiple abstraction level results and the models' incremental learning capability.

1.4 PhD publications

1. **Divya Thekke Kanapram**, Lucio Marcenaro , David Martin Gomez, Carlo Regazzoni, "Interpretable Machine learning models for abnormality detection in Ego-things network," in IEEE Internet of Things Journal, status: Submitted on 31-01-2021.

2. **D. Thekke Kanapram**, Fabio Patrone , Pablo Marin-Plaza , Mario Marchese , Eliane L. Bodanese, Lucio Marcenaro , David Martín Gómez, Carlo Regazzoni, "Collective Awareness for Abnormality Detection in Connected Autonomous Vehicles," in IEEE Internet of Things Journal, vol. 7, no. 5, pp. 3774-3789, May 2020, doi: 10.1109/JIOT.2020.2974680.

3. **Divya Thekke Kanapram**, Pablo Marin-Plaza, Lucio Marcenaro, David Martin, Arturo de la Escalera, Carlo Regazzoni, "Self-awareness in intelligent vehicles: Feature based dynamic Bayesian models for abnormality detection," Robotics and Autonomous Systems, Volume 134, 2020, 103652, ISSN 0921-8890, https://doi.org/10.1016/j.robot.2020.103652.

4. **Divya Thekke Kanapram**, Pablo Marin-Plaza, Lucio Marcenaro, David Martin, Arturo de la Escalera, Carlo Regazzoni, "Self-awareness in Intelligent Vehicles: Experience Based Abnormality Detection," Robot 2019: Fourth Iberian Robotics Conference. ROBOT 2019. Advances in Intelligent Systems and Computing, vol 1092. Springer, Cham, Online ISBN 978-3-030-35990-4, https://doi.org/10.1007/978-3-030-35990-4$_1$8.

5. M. Baydoun, D. Campo, **D. Kanapram**, L. Marcenaro and C. S. Regazzoni, "Prediction of Multi-target Dynamics Using Discrete Descriptors: an Interactive Approach," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, United Kingdom, 2019, pp. 3342-3346, doi: 10.1109/ICASSP.2019.8682272.

6. **Divya Kanapram**, Damian Campo, Mohamad Baydoun, Lucio Marcenaro, Eliane L. Bodanese, Carlo Regazzoni, Mario Marchese, "Dynamic Bayesian Approach for decision-making in Ego-Things," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), Limerick, Ireland, 2019, pp. 909-914, doi: 10.1109/WF-IoT.2019.8767204.

1.5 Limitations of scope

1. The proposed approach of learning self-awareness and collective awareness is tested using the data sets from vehicle co-operative driving task

scenarios. Therefore, the ego-things in this work are vehicles. The proposed methods are not tested in other ego-things, such as drones and other IoT devices. Therefore, it is difficult to say that the methods can be easily applicable in general settings without modifying the framework. It requires further work and modifications to make an optimized model representing individual and collaborative tasks of ego-things in general.

2. The developed dynamic Bayesian self-awareness and collective awareness models require that ego-things communicate or share all the ground truth sensory observed data. In practical applications, it could be challenging to share a massive amount of data as there are constraints for the data packet size by considering different communication protocols. Therefore, further work is required to improve the model to share only specific parameters rather than sharing all the observations.

3. The developed data-driven models are mainly used to detect abnormal situations are happening either around a single agent or in the network of agents. The work should be further focused to develop an autonomous decision-making system based on the detected anomaly. Moreover, abnormality classification should be added to improve the reliability of the system.

4. Another limitation is the criteria to select modalities that provide better abnormality detection, i.e., the requirements to choose the best data combinations among many sensory data. It is essential to develop a generalized approach to pick the sensor's characteristics and the dynamic models to have good abnormality detection.

5. It is also essential to add a module for incrementally learning of new models automatically whenever the existing models produce errors as an anomaly. However, in this work, it is proposed an initial level approach for emergent learning.

1.6 Thesis structure

The structure of the thesis is outlined below. Each chapter aims to address parts of the research questions described above. The logical progression of work is illustrated graphically in Fig. 1.3.

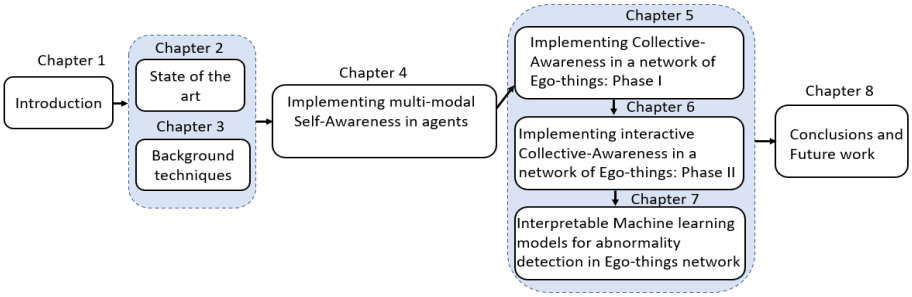


Figure 1.3: Thesis progression.

Chapter 1 - **Introduction**: This chapter explains the importance of this research work, the research questions, a summary of the main contributions, and the list of published papers during the Ph.D.

Chapter 2 - **State of the art:** This chapter discusses the background and state of the art related to the topic. At first, it highlighted the importance of the data-driven model learning approach by comparing it against the model-driven approach. Then explained the work done in the field of self-awareness in agents. Finally, the work done in the field of collective awareness in agents network have been discussed.

Chapter 3 **Background techniques:** This chapter presents the background techniques used in this work: BNs, DBNs, GNG algorithm, MJPF. Moreover, the last part of this chapter included the important definitions and concepts used in this work.

Chapter 4 - **Implementing multi-modal Self-Awareness in agents:** This chapter proposes the representation and modeling of the dynamic behavior of a single agent. DBN models are learned from sensory data (features), and a MJPF is used to make inferences on the learned DBN models and detect abnormal situations. Finally, the multi-modal abnormality detection results are presented and discussed.

Chapter 5 - Implementing Collective Awareness in a network of Ego-things: Phase I : This chapter presents the initial approach to build and test multi-modal collective awareness (CA) for a network of agents. The MJPF is employed to infer the future states of the entities and detect abnormal situations. Moreover, the performance metrics are discussed to assess the algorithm's reliability and accuracy. Furthermore, this chapter discussed the dynamic Bayesian model-based abnormality detection performances under the different channel and source conditions. The effects of distances among agents and of the delays and packet losses are analyzed in different scenario categories.

Chapter 6 - Implementing interactive Collective Awareness in a network of Ego-things: Phase II: This chapter presents the implementation and testing of a multi-modal collective awareness (CA) model for a network of ego-things where different communication protocols are considered in order to evaluate the model's performance under different parameter values and environmental conditions.

Chapter 7 - Interpretable Machine learning models for abnormality detection in Ego-things network: This chapter concentrates on developing and testing interpretable data-driven machine learning (ML) techniques to detect local and global abnormal situations in agents' networks. The proposed approach assumes that the data-driven models to be chosen should support emergent self-awareness of the agents at multiple abstraction levels. It is demonstrated that incrementally updating the learned representation of models based on the agent's progressive experiences is strictly related to interpretability capability.

Chapter 8 - Conclusions and Future work: This chapter concludes the research achievements and discusses the propositions of future research.

Chapter 2

State of the art

This chapter firstly focuses on making a comparison between data-driven and model-driven approaches covering their advantages and drawbacks. Secondly, the main state-of-the-art contributions regarding the development of self-awareness in *agents* and collective awareness in *agents'* networks are discussed.

2.1 Model driven and data driven approach

There are mainly two paradigms for solving the classification and state estimation problem in sensor data: Model-driven and Data-driven. The model-driven approach starts with a solid idea of how a physical system works. It considers the desired states or events to be detected, and then it generates a hypothesis about what aspects might be detectable from the outside and what the target signal will look like. It finds a correlation between the recorded, collected samples from a task and what it is trying to detect. Then a detector by hand finds those hard-won features out in the real world automatically. On the other hand, data-driven is a new approach enabled by machine learning. In this approach, learning algorithms from the data can spot connections and correlations not known before. Both of these approaches have their advantages and drawbacks.

Model-driven approaches are powerful because they rely on a deep understanding of the system or process and can benefit from scientifically established relationships. Models cannot accommodate infinite complexity and generally must be simplified. They have trouble accounting for noisy

Table 2.1: Model driven and data driven approaches comparison.

	Model driven : Relies on a-priori model	**Data driven: Learns the model from data**
1	Physical model	Non-physical model
2	High generalizability	Limited generalizability
3	Limited adaptability to data	Highly adaptive to data
4	Computationally demanding	Computationally efficient

data and non-included variables.

Data-driven approaches based on machine learning tools and techniques require a fair amount of data to get reliable results. Artificial intelligence (AI) tools that discover features and train-up classifiers learn from data samples, and enough data are required to cover the full range of expected variation and null cases. Some tools are powerful enough to generalize from limited training data and discover viable feature sets and decision criteria independently. Still, many machine learning approaches require truly *big data* to get meaningful results. Table 2.1 summarises the advantages and limitations of model-driven and data-driven approaches.

In [99], the authors proposed a data-driven model for generating a sketch from an input image. Then made a comparison to the existing model-driven approaches. The work, [28], proposed and compared the model-driven methods and data-driven techniques of prediction models for crop growth in interaction with their environment as dynamical systems. This thesis proposed various data-driven approaches in developing self-awareness and collective awareness in agents with a particular functionality to detect abnormal situations.

2.2 Self-awareness in agents

Artificial intelligence (AI) is the concept that allows agents/machines to perform any task autonomously in any situation. Under the umbrella of AI, applications such as machine learning, deep learning, etc., are increasingly used to implement solutions in various fields, including self-driving vehicles. The intense use of machine learning techniques applied to the sensory data helps deal with the system's uncertainty to a certain extent. Such multi-sensory data used to build models that can make predictions of the agents'

future states.

Self-awareness (SA) is a broad concept that describes a cognitive property of a biological agent. SA can be defined as an agent's quality to become its own reflective observer by processing information about the self. It occurs when an agent focuses not only on the external environment but also on the internal milieu [69]. Over many years, self-awareness has been studied in multiple research disciplines, such as cognitive sciences, psychology, and philosophy [8, 7, 9, 95]. Moreover, according to the definition in [34], the circumstantial cues remind the agents of themselves and lead to give more attention to self and away from the environment. On the other hand, [44] proposed the idea of private and public self-awareness. Authors of [34, 44], examine the impact of private self-awareness in decision making. The self-awareness concept is widely studied in biology, which has been reproduced in artificial systems to enrich the capability of autonomy in different fields, including machine learning and robotics [85, 101]. The main challenge in most of these approaches is how self-awareness capabilities integrate into artificial agents.

An artificial agent can be considered self-aware if it can dynamically observe itself and surrounding environment through different exteroceptive and proprioceptive sensors and learn and maintain a contextual representation by processing the observed multi-sensorial data [84]. Developing self-awareness in artificial agents may reduce human efforts in different areas, and in some fields, the human operator can be entirely replaced by machine intelligence.

In [17], the authors propose an approach to develop a multilevel self-awareness model by focusing on one agent. The developed self-awareness approach is learned by using multisensory data of a vehicle normally interacting in an environment. The model allows the agent to become able to detect abnormal situations present in its surrounding environment. The learning process of the self-awareness model for autonomous vehicles based on data collected from human driving is described in another work [83]. Other related works in this direction aim to enrich the experience of cooperative and secure driving [63, 76].

In this thesis, dynamic Bayesian networks (DBNs) learned using sensorial data are considered. The data is recorded during training experiences as generative models capable of allowing a SA agent to predict states in

future similar testing experiences. Additional probabilistic inference features related to DBN models allow the SA of the agent to detect possible abnormalities in new experiences. Prediction, estimation, and abnormality detection are the emergent (i.e., data-driven) SA features discussed in this thesis as a collective property of a set of agents aiming to perform the same task.

Developing self-awareness in agents has been shown to increase agent confidence when executing autonomously tasks and explainability of its own actions in terms of emergent SA models learned. Human efforts in different areas can take advantage, and in some fields, this can increase confidence in autonomous systems, allowing the human to reduce its overcontrol work. An intelligent vehicle [IV] [22] can be seen as a straightforward example of an agent: it firstly perceives information from the surrounding environment. It then uses obtained information to make decisions autonomously in different situations. However, this does not always imply that the intelligent vehicle can explain to itself and the human user the causal sequence of events that carried it to make decisions. Self-awareness addresses within AI the set of techniques/models that allow agents/machines to describe the relationship between perceptions and actions the agent has to do to perform a task. In this context, machine learning and deep learning, etc., are increasingly used to obtain SA models in a data-driven way, and self-driving vehicles [20] can benefit from such methods. Machine Learning techniques capable of dealing with uncertainty to learn the SA model from multisensory signals coming from the vehicle's sensors are particularly useful. Such models can be of the generative type, allowing predictions of future or lower-level states of the agent in consideration to be made when analyzing new data sequences.

Research in intelligent and autonomous vehicles occupies a prominent place in intelligent transportation systems (ITS) in recent years. In [15], the authors propose an approach to develop a multilevel self-awareness model learned from an agent's multisensory data. Such a learned model allows the agent to detect abnormal situations present in the surrounding environment. In another work, [83], the learning of self-awareness models for autonomous vehicles is based on the data collected by different maneuvering tasks performed by a human driver. In [83], the visual perception and position data are used as different modalities, and the cross-correlation between modalities is analyzed for detecting abnormal situations. On the other hand, in

[25], the authors propose a new architecture for mobile robots with a model for risk assessment and decision making when detecting difficulties in the autonomous robot design. In [103], the authors proposed a model of driving behavior awareness (DBA) that can infer driving behaviors such as lane change. In [51], an approach to detect abnormalities in dynamic systems by the models learned from the different features of an *agent* is presented. Moreover, it examines the most precise model to detect abnormal situations. However, it involves one agent and doesn't highlight the issue of co-operative driving.
In most of the related works, either the data from one entity is used, or the objective was limited; for example, in [103], the aim was to detect lane changes left or right side of the considered vehicles.
The first part of this thesis used the real vehicle data to build multi-modal data-driven switching DBN models, and, finally, a performance comparison is made among the models.

2.3 Collective awareness for an agents' network

Collective awareness is an extension of the self-awareness concept to a network of ego-things that co-operate to perform a given task with different interdependent roles. Collective awareness allows the network to understand whether the perception-action information processing models agents are provided with allow them to predict the dynamic evolution of the current situation, as well as to coherently detect global anomalies in a distributed way [53].

Studies have been conducted in the field of *collective awareness* or *collective consciousness*. The collective consciousness was a term coined by the French sociologist Émile Durkheim (1858–1917) to refer to the shared beliefs and moral attitudes that operate as a unifying force within society [73]. Collective consciousness or collective awareness plays a significant role when a group of agents needs to perform a task by co-operating and communicating together to achieve common or individual goals. In [24] the authors investigated the key requirements to achieve collective action in decentralized community energy systems (dCES); collective awareness to enhance the sense of collective responsibility, social networking to promote self-organization.

Each *agent* in the multi-agent system can take autonomic actions to a certain extent along with the ability to interact with other agents [102]. A group of such self-aware agents can form a network that has collective awareness capabilities. With such an ability, each agent in the system should be aware of itself and other agents' activities. Distributed state estimation and tracking are fundamental collaborative information processing problems in wireless sensor networks (WSNs).

Multi-sensor fusion and tracking problems have a long history in signal processing, control theory, and robotics [6, 12, 11, 36]. For instance, in [36], the authors presented a survey of Bayes filter implementations and showed their application to location-estimation in real-world tasks common in pervasive computing. Moreover, the distributed state estimation issues in wireless networks with packet-loss have been the center of much attention lately [92, 47, 87]. There has been a significant development in the study of Kalman filtering in the presence of data packet drops [61, 68, 79, 87]. In [61], the authors proposed distributed estimation algorithms for linear time-varying systems by considering packet drops in the observed data sequence. In addition to that, numerical examples are provided to verify the effectiveness of the proposed filter.

The recent advances in the WSN technology also boost the study of a distributed Kalman filter (DKF) [5, 4, 74, 94], where each sensor node in the WSN can compute local estimates via Kalman filtering based on its own observations and the information sent from its neighboring sensors. The paper [74] provided a formal derivation of the optimal Kalman-consensus filter (KCF) and a performance comparison made between the proposed suboptimal Kalman-Consensus Filter and the previous distributed Kalman filtering (DKF) algorithms.

The existing literature either focused on the local estimation of agent states by a distributed Kalman Filter or state estimation under the data packet loss by sharing among agents'. It clearly shows that the lack of a model represents the group of agents' interaction and behavior globally and underlying reasons in different situations.

This work has developed multi-modal collective awareness (CA) models by considering exteroceptive and proprioceptive sensory data from networked agents. Each of the considered modalities extracts different system features that enrich contextual awareness to detect abnormalities at different

abstraction levels. The main features of the developed model wrt existing literature are listed below:

1. Each of the considered modalities extracts different system features that enrich contextual awareness to detect abnormalities at different abstraction levels. Different probabilistic distance metrics are used in the MJPF to estimate the abnormality. To estimate continuous level abnormlaity, the filter's innovation and Hellinger distance metrics are used. For the discrete level anomaly estimation, Kullback Leibler (KL) divergence, which calculates the probabilistic distance between two distributions, is considered.

2. The model's significant feature is that it is data-driven, i.e., different abstraction levels of the model have been learned from multi-sensory data.

3. The effects of wireless communication channels on the model performance have been analyzed by considering different channel conditions and communication protocols. Then compared the obtained results by different performance evaluation metrics.

4. The developed CA model is interpretable; anomaly detection results have been utilized to learn new models incrementally to represent the situation. The model can also predict the future states better in case of similar situation occurs. The interpretability feature will help to increase the robustness and reliability of the system.

2.4 Concluding remarks

This chapter firstly presented a comparison between data-driven and model-driven approaches. In this thesis, data-driven approaches are used to learn and test self-awareness and collective awareness functionality in agents' networks. Then made a study and analysis of existing literature on artificial agents' self-awareness and collective awareness. Finally, it highlighted the importance of this research against existing literature.

Chapter 3

Background techniques

This chapter explains the major techniques used in this work, such as Bayesian networks (BNs), dynamic Bayesian networks (DBNs), growing neural gas (GNG), and Markov jump particle filter (MJPF). Moreover, the definitions and concepts used in this work are explained in this chapter.

3.1 Bayesian networks (BNs) and dynamic Bayesian networks (DBNs)

BNs are directed acyclic graphs (DAGs) in which the nodes represent variables, and the arcs (edges) signify the existence of direct causal influences between the linked variables. The strength of these influences are expressed by forward conditional probabilities [78]. For any given edge between the variables (nodes), if there is a causal relationship between the variables, the edge will be directional, leading from the cause variable to the effect variable. BNs can be defined as a special case of a more general class called graphical models in which nodes represent random variables, and the lack of arc represents conditional independence assumptions between variables. The vertices of the BN are called nodes and are represented as circles containing the variables. The connections between the nodes represented by arrows are called edges or arcs and represent dependence between the node variables.

Fig. 3.1 illustrates examples of BNs encoding different types of dependencies between three variables. The node where the arc originates is called the parent, while the node where the arc ends is called the child. For exam-

ple, in Fig. 3.1 (a), X is a parent of Z, and Y is a child of Z.

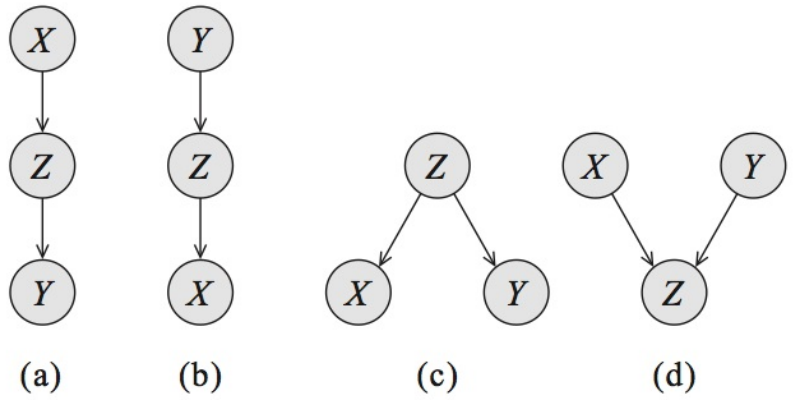


Figure 3.1: Bayesian networks over three variables, encoding different types of dependencies: (a,b) cascade, (c) common parent , and (d) v-structure.

Bayesian networks are a type of probabilistic graphical model that uses Bayesian inference for the computation of probabilities. The probability theory is based on three basic axioms:

1. $0 \leqslant P(X) \leqslant 1$
2. $P(X) = 1$ if and only if X is certain, and
3. If X and Y are mutually exclusive, then $P(X \bigcup Y) = P(X) + P(Y)$, and and a fundamental rule of probability calculus: $P(X, Y) = P(X/Y)P(Y)$ where $P(X/Y)$ is the probability of the joint event $X \bigcap Y$. This brings us to the Bayes' rule for computing posterior probability: $P(X/Y)$, given the prior one $P(X)$, and the likelihood $P(Y/X)$ that Y will materialize if X is true: $P(X/Y) = P(Y/X)P(X)/P(Y)$. Here X represents hypothesis, Y represents evidence, and $P(Y)$ denotes the normalizing factor.

BNs can be thought of as a knowledge base [91] which explicitly represents our beliefs about the elements in the system and the relationships that exist between these various elements of the system. Such a knowledge base aims to infer some belief or make conclusions about some processes or events in the system.

Similarly, a DBN is a BN which relates variables to each other over adjacent time steps [96]. DBNs are usually defined as a special case of singly

connected Bayesian networks specifically aimed at time-series modelling like stated in [77]. All the nodes, edges and probabilities that form a static interpretation of a system is identical to a BN. In the case of DBN, the nodes can be denoted as the states, because they include a temporal dimension. The states of any system described as a DBN satisfy the Markovian condition, that is defined as follows: The state of a system at time k depends only on its immediate past, i.e., its state at time *k-1*. Also, this property is frequently considered as a definition of first-order Markov property as in [71]: the future is independent of the past given the present. Fig. 3.2 shows the representation of a DBN with the variables at two different time steps, k-1 and k. The next time step, k, is dependent on time step k-1. There are two types of edges (dependencies) that can be defined in a DBN such as *intra-slice* topology (within a slice) and *inter-slice* topology(between two slices). In Fig. 3.2, a_0, b_0, c_0 are initial states and a_i, b_i, c_i are future states where i=1,2,3,. . . ,n. The probability distribution for this DBN (refer Fig. 3.2) at time k is:

$$p(x_k/x_{k-1}) = \prod_{i=1}^{N} p(x_k^i/Pa(x_k^i)) \tag{3.1}$$

where x_k^i is a node at time instant k and $Pa(x_k^i)$ are the parent nodes of x_k^i

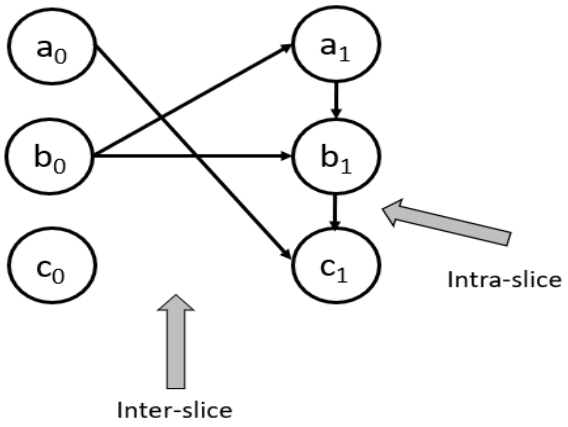


Figure 3.2: DBN representation with two-time steps.

To decide the relation between two variables in a DBN, whether intra-slice or inter-slice, depends on how tight the coupling is between them. If the effect of one variable on the other is immediate (much shorter than the

time granularity), the influence should manifest as an intra-slice edge. On the other hand, the influence should show as an inter-slice edge if the effect is slightly longer term. An inter-slice edge connecting two instances of the same variable is called *persistence-edge*. Hidden Markov models (HMMs) and Kalman filter model (KFM) are specific non-trivial examples of DBNs. HMMs are formed by one hidden variable with persistence links between time-steps and one observed, as shown in Fig. 3.3. The probabilities in HMM can be defines as follows:

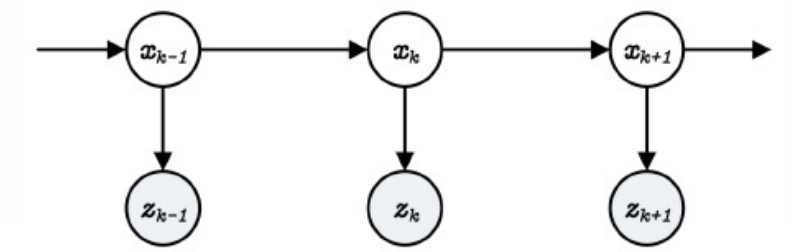


Figure 3.3: Hidden Markov Model (HMM).

- $P(x_0)$ is the initial state distribution and represents the uncertainty on the state's initial value.
- $P(x_k/x_k - 1)$ defines the transition model. It tells how the states evolve in time.
- $P(z_k/x_k)$ is the observation model. It represents how the observations are related and generated by a hidden state. It is also called the likelihood.

On the other hand, KFM is characterized by one continuous hidden node. All nodes are assumed to be Linear-Gaussian distributions. The probabilities in KFM can be defined as:

- $P(x_0) = N(x_0, Q_0)$ is the initial state distribution.
- $P(x_k/x_k - 1) = N(Fx_k - 1) + G(u_k, Q)$ defines the state transition model (u represents the control term and Q is the process noise covariance).

- $P(z_k/x_k) = N(H_x, v)$ is the observation model (H is the observation matrix and v is the observation noise). It represents how the observations are related and generated by a hidden state. It is also called the likelihood.

3.2 Growing Neural Gas (GNG) algorithm

In this work, the GNG algorithm is used to cluster the input state space to find the topological structures that closely reflect the structure of the input distribution.

The purpose of cluster analysis is information retrieval (data-mining) of a huge amount of data. Therefore, clustering-algorithms use to split the data into clusters, and the feature of the data contained in each cluster has a high degree of similarity. In general clustering-algorithms map a set of N input vectors:

$\mathbf{X} = \{x_1, x_2, ..., x_n | x_i\} \in R^d, i = 1, 2, ..., N$ into c clusters. Whereby 2 $\leq c \leq$N.

$\mathbf{X}$ represents the input states and c is the number of clusters.

In general, clustering can be described as the process of organizing a collection of k-dimensional vectors into groups whose members share similar features in some way. A k-dimensional vector represents each group, called a code vector (other names used are centroid and node). There are many algorithms available for clustering: K-means [64], self organising map (SOM) [57], neural gas (NG) [67], growing neural gas (GNG) [39], density-based spatial clustering of applications with noise (DBSCAN) [35], etc. The SOM algorithm can compress large multidimensional datasets into a fixed number of representative units. However, the dimension of the representative units (clusters) needs to be defined before, and it may sometimes cause not intuitive for representing the characteristics of data structure.

In contrast to the SOM, GNG is an unsupervised, adaptive, and incremental neural network that learns topologies; it grows during the learning process and does not require users to define the number of representative units called nodes beforehand. Such a dynamic property is an advantage over other clustering algorithms for using it in many applications. DBSCAN is a

density-based clustering algorithm that finds a number of clusters starting from the estimated density distribution of corresponding nodes. Although it has many advantages, such as discovering arbitrarily shaped clusters and robust detection of outliers, the algorithm sometimes fails to identify clusters in those situations of varying density of considered data or if the dataset is too sparse. The dataset considered in this work is sparse and multidimensional, so that we have chosen the GNG algorithm by considering its advantages over other clustering algorithms.

The GNG algorithm [39] extends the NG algorithm by adding a local error measure for each node. This error is accumulated based on the node's distance to the input samples presented to the network. Regularly, a new node is inserted between the two nodes that have accumulated the largest amount of error. The GNG network's underlying graph starts with two nodes and expands until it reaches a predefined maximum number of nodes. Therefore, the number of nodes in the GNG is no longer fixed as it is in the NG algorithm. A second addition proposed by [38], is the utility measure. It is used to determine each node's usefulness by estimating the increase of the global network error if the node would not be present. Based on their utility value, the nodes contributing little to reducing the global network error can be removed. GNG incrementally creates a network of nodes, given some input distribution in R^n. GNG can be used for vector quantization by finding the code-vectors in clusters. In GNG, these code vectors are represented by the reference vectors (the position) of the GNG nodes. It can also be used for finding topological structures that closely reflects the structure of the input distribution. GNG has been widely used in recent years for different applications, mainly for clustering or topology learning. In this work, GNG is used to segment data collected by various sensors into a set of regions that facilitate a semantic interpretation of data and defines local linear models employed for prediction purposes.

In Section 4.1.1, the pseudo-code for the GNG algorithm is presented with examples.

3.3 Markov Jump Particle Filter (MJPF)

In this phase, a probabilistic switching model called MJPF [33, 51] has been chosen to make inferences on the DBN models learned in the training phase.

The filtering algorithms like MJPF [48, 31] and IMM filters [75] allow an agent to predict and estimate target motion according to multiple probabilistic models. The filters differ in the inference methods they use to perform prediction and update steps. While MJPF uses particle filters at discrete levels together with Kalman filters at continuous levels, IMM filters can use different approaches. For example, in IMM filters [23, 106], a model-driven approach is performed to fuse Kalman filters. In general, IMM filters can be coupled with parameter estimation learning methods specific to the inference approach used that can be used on training sequences. However, parameters are often chosen by design, and fixed discrete state transition probabilities are provided offline from the discrete variables switches' frequency. The number of models is generally a priori fixed, limiting the descriptors of the agents' dynamics. In this work, we used MJPF, a type of Markov jump linear system (MJLS) that uses a parametrized couple of Particle filter and Kalman filters that can be learned from data. This allows as in IMM inferences on a Dynamic Bayesian Network jointly at continuous and discrete levels. However, the data-driven approach used in this work is based on a free energy minimization approach that allows a varying number of dynamic models to be estimated together with temporal transition probabilities that characterize the models' discrete temporal evolution.

In IMM, model switching is mainly dependant on a time-independent transition probability matrix; in MJPF here used, co-occurrence probability and transition models learned are time-dependent, so allowing a time-variant transition probability, specific for each dynamic model, to be employed. In MJPF, the number of particles employed at the discrete level is proportional to the dynamic models. The method explore in parallel an alternative set of dynamic model predictions by evaluating the best choices depending on anomaly detection capabilities added.

MJPF is a switching model that describes an object's behavior changing from time to time: the object follows one linear behavior and then switches to another behavior. In this work, MJPF is applied to learned DBN as shown in Fig. 3.4. The learned three level DBN model is linear with Gaussian noise to relate discrete and continuous variables, predicting together the next super state and state. Switching discrete variables describe a set of behaviors, and such behaviors can be learned from the data. For each region, we can have different behavior and model to predict the movement of the

object. In MJPF, use KF in state-space and PF in super state space. Such filters are used together to make a Bayesian inference. They predict and estimate the states of a moving agent.

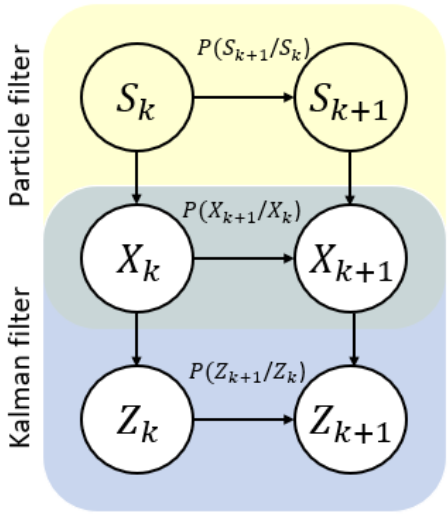


Figure 3.4: Three level DBN model representation.

- Kalman filter (KF): KF is an iterative filter for linear systems with Gaussian noise. It needs a definition of dynamic model

$$x_{k+1} = Ax_k + BU_{sk} + n_k, \tag{3.2}$$

and observation model

$$y_k = AHx_k + \omega_k, \tag{3.3}$$

where k is time step, x_k is the state, and y_k is observation, n_k and ω_k are zero mean prediction noise and measurement noise, independent of each other, white and with normal probability density functions with covariance matrices Qs_k and R [90].

A is the state transition matrix that maps the agent's position as constant with respect to the previous state. B is a control input model that maps the action of an agent to the following states. For each superstate nodes in DBN (yellow shaded area in Fig. 3.4), we use different control vector Usk.

Kalman filter works with two steps:
- prediction finds $p(x_k/Z_{k-1})$

$$\hat{x}_{k/k-1} = Ax_k + BU_{sk} + n_k, \tag{3.4}$$

$$P_{k/k-1} = AP_{k-1/k-1}A^T + Q \tag{3.5}$$

- update finds updated estimate with measurement z_k, $p_{(xk/Zk)}$ and updated covariance matrix $P_{k/k}$:

$$z_{k/k-1} = H\hat{x}_{k/k-1} + m_k \tag{3.6}$$

$$\hat{x}_{k/k} = \hat{x}_{k/k-1} + K_k(z_k - H\hat{x}_{k/k-1}) \tag{3.7}$$

$$P_{k/k} = P_{k/k-1} - K_k S K_k^T = [I - K_k H]P_{k/k-1} \tag{3.8}$$

$$K_k = P_{k/k-1}H^T(HP_{k/k-1}H^T + R)^{-1} \tag{3.9}$$

where I is identity matrix, $\hat{x}_{k/k-1}$ is a priori estimate of the state x_k given measurements until $k-1$, $\hat{x}_{k/k}$ is a posteriori estimate of the state x_k given measurements up to time k, K_k is the Kalman gain, $P_{k/k-1}$ is covariance of a priori estimate and $P_{k/k-1}$ is covariance of a posteriori estimate.

The difference $v_k = z_k - H\hat{x}_{k/k-1}$ in Eq. 3.7 is called the measurement innovation or the residual. Residual compares predicted states $H\hat{x}_{k/k-1}$ and actual measurement z_k. If they are very far, observation is used to have a better estimate of the following state.

From Eq. 3.10 we can observe that when measurement error covariance R is small, the gain is high, and the residual has more weight in Eq. 3.7. The actual measurement z_k is more reliable than the prediction. On the other hand, when $P_{k/k}$ is small, the Kalman gain doesn't weigh residual much, so z_k is trusted less [100].

- Particle filter (PF): PF can be important in the cases when systems are non-linear, and noise is non-Gaussian. In our case, posterior probability density function, $p(x_k/Z_k)$ estimated at every step isn't Gaussian. It is a non-parametric probability density function approximated with a set of particles in different positions state-space, each particle with a different weight.
  Particle filter with sequential importance resampling (SIR) algorithm uses as importance function dynamic model $p(x_k/x_{k-1})$ to select particles. At the beginning of each new iteration, the weight of each particle is set to $1/N$. Then the weights are updated using measurement to say how much probable is each particle, considering $p(z_k/x_k)$ and the values are normalized. At each step in the resampling phase, the particles with low weights will be deleted, and those with large weights will be multiplied to have the same number of particles as before.

  The MJPF predicts both state and superstate by estimating a posterior probability density function as below:

$$p(s_k, x_k/Z_k) = p(x_k/s_k, Z_k)p(s_k/Z_k) \tag{3.10}$$

  $p(s_k, x_k/Z_k)$ can be estimated by a PF. Particles move in superstate space and are drawn from a proposal function $q = p(s_k/sk-1)$. Therefore, at the next time step, we predict the particle from the transition matrix. A different Kalman filter is associated with each particle s_k^*, different for each discrete zone s_k, to connect the discrete state to the continuous state and to make predictions and update of state. Each filter is described by a different dynamic model and by a unique shared observation model.

  The different versions of MJPF are explained in the next chapters, where it is used to make inferences on real data observations.

3.4 Definitions and concepts

This section includes some of the definitions found in the existing literature related to the concepts used in this work.

- **Ego-thing**: Ego-thing can be defined as intelligent autonomous entities that can perceive their internal as well as external parameters and adapt themselves when they face abnormal situations [51]. In this thesis, ego-thing, agent, object, and vehicle are used as synonymous.
- **Self-awareness (SA)**:
  - SA can be seen as the capacity to become the object of one's own attention. It occurs when an organism focuses not only on the external environment but on the internal self ; it becomes a reflective observer by processing private and public self-information [69].
  - SA is a capability of an autonomous system to describe the acquired knowledge about itself and its surroundings with appropriate models and learn new models incrementally when it comes to new experiences [84].
- **Collective awareness (CA)**: The term collective consciousness was coined by the French sociologist Émile Durkheim (1858–1917) to refer to the shared beliefs and moral attitudes that operate as a unifying force within society [73]. Collective consciousness or collective awareness plays a significant role when a group of agents needs to perform a task by co-operating and communicating to achieve collective or individual goals. In [25] the authors investigated the key requirements to achieve collective action in a decentralized community energy systems(dCES); collective awareness to enhance the sense of collective responsibility, social networking to promote self-organization.

  Collective awareness is an extension of self-awareness concept to a network of ego-things that cooperate to perform a given task with different interdependent roles. Collective awareness allows the network to understand whether perception-action information processing models they are provided of allow then to predict the dynamic evolution of the current situation, as well as to coherently detect global anomalies in a distributed way [53].
- **Multimodality**: Different sensor modalities can be used by an agent to collect information by its own sensors about its own state

(proprioceptive) and context one (exteroceptive); consequently, collective awareness makes it necessary to be capable of learning models from heterogeneous sensor modalities. The capability to estimate causal dynamic connections of generalized variables related to different modalities is a key aspect to allow agents to be provided of collective awareness models related to co-operative tasks they have to perform.

– **Interpretability**: In this thesis, interpretability is defined as the capability to use anomaly data to modify the existing model to make the future inferences more accurate.

Chapter 4

Implementing multi-modal Self-Awareness in agents

This chapter proposed a novel approach to developing multi-modal self-awareness in agents to detect abnormalities. Several features of the observed sensory data used learning the multiple inference self-awareness models. The method is evaluated with real dataset from a moving vehicle performing different tasks in a closed environment. The proposed method can be useful for selecting relevant features when dealing with numerous features in networking operations.

This chapter's work is highly motivated by models proposed in [14, 17]. In [17], the authors proposed a method to develop a multi-layered self-awareness in autonomous entities and exploit this feature to detect abnormal situations in a given context. On the other hand, [14] introduces a MJPF, which consists of continuous and discrete inference levels that are dynamically estimated by a collection of KFs assembled into a PF algorithm. MJPF enables the prediction of future states in continuous and discrete levels. Most of the related works [49, 83] use position-related information to make inferences. However, the information related to the control of an autonomous entity plays a significant role in predicting future states and actions of the entity. Accordingly, it is imperative to consider variables related to control to develop predictive models for making the system more efficient.

The novelties of this chapter are listed as follows:

1. A method proposed to learn data-driven self-awareness models in agents. Low dimensional multi sensory data containing information of actuator are divided into different sets of data, generating multiple inference subsystems.
2. The work includes a novel strategy for segmenting and interpreting clusters of generalized errors. To obtain the clusters, used a growing neural gas (GNG) algorithm.
3. The proposed strategy selects the set of sensory data (features) that optimize the prediction of future states and the detection of abnormalities. A probabilistic anomaly measurement based on the Hellinger distance is used for the anomaly estimation.

4.1 Proposed methodology

This section firstly describes how the "awareness" can be modeled into a *thing/agent* that can become an "ego-thing". Ego-things can be defined as intelligent autonomous entities that can perceive their internal as well as external parameters and adapt themselves when they face abnormal situations.
In general, an ego-thing can be equipped with various *exteroceptive* and *proprioceptive* sensors. Proprioceptive sensors measure values internally to the agent, e.g. battery level, wheel position, joint angle, etc. These sensors can be encoders, potentiometers, gyroscopes, compasses, etc. Exteroceptive sensors are used for the observation of the environments, objects. Sonar sensors, camera, IR sensitive sensors, ultrasonic distance sensors are some examples of exteroceptive sensors. In this work, we mainly considered the control data (i.e., steering angle, rotor velocity and rotor power) of the ego-thing to develop self-awareness. The collected sensor data have been initially synchronized to match their time stamps and then categorized into groups. In addition to that, the sensor datasets have been normalized to bring the numeric columns in the datasets to a common scale by not distorting the differences in the ranges of values or losing information. The proposed method can be divided into two parts: *offline training* and *online testing*. The purpose of the offline training phase is to learn new DBN

models from the dynamic data series collected from the reference situation task. It starts from a set of data series and learns a *vocabulary*, transition probability matrices, and finally, we obtain the dynamic models. On the other hand, in the online testing phase, a filter called Markov jump particle filter (MJPF) is applied on the learned switching DBN models by giving a set of new dynamic data series as input. Switching DBNs are probabilistic models to integrate observations received from multiple sensors in order to understand the environment and take appropriate actions. The output of the switching DBN model is the future state predictions and abnormality measurements.

4.1.1 Offline training phase

In the training phase, the ego-thing will learn a number of switching dynamic Bayesian network (DBN) models from the data collected by the sensors mounted on its body. DBNs are probabilistic models used to integrate observations received from multiple sensors allows a system to understand the environment and take appropriate actions. Moreover, such models allow a system to understand temporal relationships and analyze time series, a set of sequential observations in time. The system can make inferences to estimate the probability density functions of unknown states from given ground truth observations and initial probability distribution. In this work, ego-thing is autonomous vehicle, as shown in Fig. 4.8a.
A block diagram representation of the training phase is shown in Fig. 4.1, and the steps followed to learn the switching DBN models are explained below.

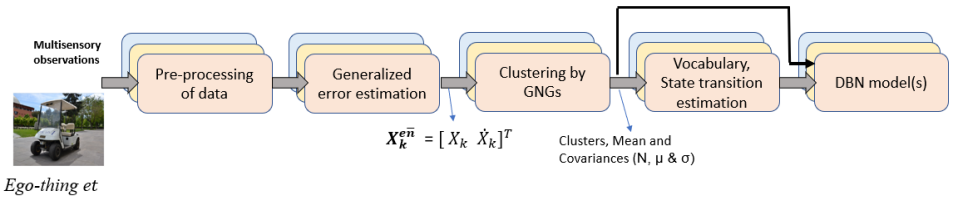


Figure 4.1: DBN model learning (training phase).

– **Data preprocessing and state estimation**

Firstly, a set of observations of control data variables are acquired. Then prepossessing techniques are applied to the data, for example; synchronization, filtering, and smoothing operations performed to remove outliers and irregularities present in the datasets.

* Data synchronization: The aim of the synchronization operation is to make a match of the timestamps of data collected by different sensors. Firstly, calculate the difference between timestamps of datasets collected from two different sensors. Then, match the data samples with minimum difference in timestamp values.
* Filtering: A mean filter was used to perform a filtering operation on data. In this filter, the outlier is defined as points outside three standard deviations from the mean. The outlier is replaced with the nearest element that is not an outlier. The default window for mean calculation is five-element window.
* Smoothing: A moving average filter is used to perform a smoothing operation on data. In this filter, the window size can be changed to adjust the smoothing. In this work, default window size is used, i.e.,window size of five.

Additionally, the datasets have been normalized. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. From here onwards, we focused only on the control variables combination steering angle-rotor power $(S-P)$ to explain the DBN model learning process in detail, and the same methodology applied to other modalities for learning models.

Let Z_k^m be the measurements of control variables (i.e., steering angle s and rotor power p) in the ego-thing et at the time instant k and X_k^m be the state associated to the measurement Z_k^m, such that $Z_k^m = g(X_k^m) + \omega_k$. The function $g(\cdot)$ is assumed to be linear, and it maps states into observations. At the same time, ω_k represents the noise associated with the sensors. Assuming the function to be linear allows the work to focus on non-linearities in the learned dynamic models and relates to the agent's capability

to predict future states and anomaly detection. As explained in [37], states' time derivatives allow predictive models to make inference dynamically even when some unknown quantities are constant.

The generalized error (GE) can be defined as the error that consists of coupled information such as a state and the higher-order derivatives of states. An initial reference generalized filter [27] could be applied to low dimensional sensory data to produce generalized error. Each modality data sequence has been used to produce the generalized error from which to learn the task model.

The generalized error of data combination m for the ego-thing et can be written as:

$$\boldsymbol{X}_k^m = [X_k^m \ \dot{X}_k^m \ \ddot{X}_k^m \cdots X_k^{(L)m}]^\intercal, \tag{4.1}$$

where (L) indexes the L-th time derivative of the state.

The state related to the observations can be be written as :

$$X_k^m = \begin{bmatrix} s \\ p \end{bmatrix}$$

The l-th time derivative in et at the time k can be approximated as:

$$X_k^{(l)m} = \frac{X_k^{(l-1)m} - X_{k-1}^{(l-1)m}}{\Delta k}, \tag{4.2}$$

where $X_k^{(0)m} = X_k^m$ and Δk is the uniform sampling time for all multi sensory data. In this chapter, the derivatives inside the generalized error (GE) are limited to 1, i.e., only the states and its first-order derivatives are considered. Once generalized error is obtained, the next step is to learn the discrete level of DBN as explained below.

– **Clustering by Growing Neural Gas (GNG) algorithm**

In order to learn the discrete level of the DBN i.e. the *yellow* shaded area in Fig. 4.6, it is required to map the generalized error (GE) into a set of nodes. Each node encodes the dynamics of

generalized error that share a common objective. To group these GE and to obtain a set of nodes, we have used an unsupervised clustering approach called GNG [39] .

The GNG algorithm is used for clustering the data to make discretization of state space. GNG is an unsupervised incremental clustering algorithm able to segment generalized states of the entity into a set of regions that encode the dynamics and expected behaviors of the variables considered. Clustering can be described as the process of organizing a collection of k-dimensional vectors into groups whose members share similar features in some way. Each such group is represented by a k-dimensional vector called a *code vector* (other names used are *centre* and *node*). The goal of clustering is to reduce significant amounts of raw data by categorizing smaller sets of similar items.

As described above, the GNG algorithm is used in this work to obtain regions from GEs produced as outputs by the initial filter, i.e., sequences of coupled state estimations and errors. Thus, GNG clusters correspond to coupled compact regions of state points and errors. Derivative errors cluster encode the description of the expected dynamics that caused the data series to vary instead of following the hypothesis of the initial filter. Different ways of changing are coded as behaviors that have been found in a corresponding compact state region. The compact state region represents switching variables in the hierarchical DBN. GNG is not the unique possible clustering algorithm that could have been employed. K-means clustering [64], self organizing map (SOM) [56], neural gas (NG) [67], etc are other possible choices. In comparison to K-means and SOM, NG converges faster, and also it has other advantages. The GNG algorithm is an improved version of the NG algorithm. In comparison to NG, it does not need any dynamically modifiable parameters. The GNG algorithm extends the NG algorithm by adding a local error measure for each node. A second addition here used, first proposed by [38], is the utility measure. By considering all the advantages mentioned above, we chose to use the GNG algorithm in this work.

The apriori defined parameters for GNG used in this work, and

the pseudo-code of different steps involved in the clustering process are explained below.

A priori defined parameters for GNG: [39]

In contrast to the NG algorithm, the parameters in GNG are not modified dynamically. The following parameters have to be determined before launching the algorithm: N, ϵ_b, ϵ_n, a_{max}, λ, α, δ.

N: represents the maximal number of nodes

ϵ_b , ϵ_n : describe the mobility of the winner nodes and its neighbours and we have used the values $\epsilon_b = 0.2$ and $\epsilon_n = 0.006$.

a_{max} : describes the maximal age of a connection and every λ iteration a new node will be created.

α : represents the reduction of error counter by inserting a new node and the value is 0.5.

δ: will reduce the overall value of the error counter every iteration step and the value is 0.995.

Pseudo-code for GNG algorithm [39]

Input: data
Result: set of nodes and their prototypes
1. Initialise two randomly positioned nodes
2. **While** there is a datapoint to proceed **do**
3. Get the next data point in the input data stream
4. Find the nearest node and second nearest node from the data point
5. Increment the age of all the edges emanating from nearest node
6. Update the edges of the nodes
5. **If** the number of datapoints passed is an integer multiple of a parameter λ **then**
6. Insert a new node (elaborate)
7. Delete each isolated node
8. Finally, decrease the error of all nodes

Figure 4.2: GNG pseudocode.

GNG algorithm example

We have considered the 2D dataset of control variables combination, steering angle-power $(S - P)$ to explain the main processes involved GNG algorithm. The dataset has been normal-

ized. Therefore, the values will be between zero and one.

1. **Initialization:** Create two nodes $s1$ and $s2$ with related randomly initialized weight vectors w_{s1} and $w_{s2} \in \mathrm{R}^n$.
   where R^n refers to the Cartesian product of n copies of the set of all real numbers.
   Also, matrices are needed to store connection information and connection age information. In addition to that, each node has a counter (error counter *f*), representing the cumulative error.
   $Node1(w_{s1}) : [0.6324 \;\; 0.0975]$
   $Node2(w_{s2}) : [0.2785 \;\; 0.5469]$

   As stated before, the node values were chosen randomly.
2. **Sampling and Matching:** Draw an input vector ζ from the input data sequence and locate the two nodes with the minimal distance toward ζ with: $s_1 = min_{w_i \in N}(||\zeta - w_i||) \wedge s_2 = min_{w_i \in N \setminus \{s1\}}(||\zeta - w_i||)$ where s_1 is the nearest node and s_2 is the second nearest node to the input vector.
   Increment the age of all connection emanating from s_1. Also sum the error counter with $\delta error(s_1) = ||w_{s1} - \zeta||^2$.

   Input vector, $\zeta = [0.3587 \;\; 0.0041]$

   Distance between input vector and the nearest node $s1$ is $d1 = \sqrt{(0.3587 - 0.6324)^2 - (0.0041 - 0.0975)^2} = 0.2892$
   Distance between input vector and the nearest node $s2$ is $d2 = \sqrt{(0.3587 - 0.278)^2 - (0.0041 - 0.5469)^2} = 0.5487$
   $\delta error(s_1) = || - 0.1803||^2 = 0.0325$
   where $d1$ and $d2$ are the euclidean distance between the input vector ζ with the node $s1$ and node $s2$ respectively.

3. **Adaptation:** Compute the displacement of s_1 and its direct topological neighbours in the direction of ζ with $\delta w_{s1} = w_{s1} + \epsilon_b(\zeta - w_{s1}); \delta w_n = w_n + \epsilon_n(\zeta - w_n)$. Whereby $\epsilon_b \;\; \epsilon_n \in (0; 1]$ represents parameters for adjusting the movement.

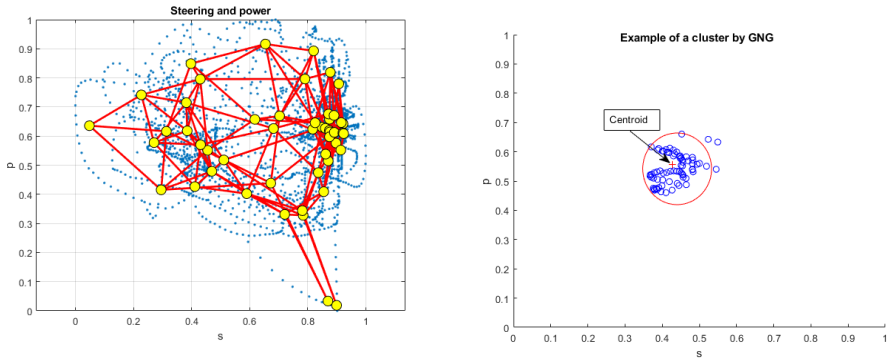
$\delta w_{s1} = [0.6324\ \ 0.0975] + 0.2 * [(0.3587\ \ 0.0041) - (0.6324\ \ 0.0975)]]$
$= [0.6324\ \ 0.0975] + 0.2 * [-0.2737\ 0.0934]$
$= [0.6324\ \ 0.0975] + [-0.0547\ 0.0187] = [0.5776\ \ 0.0788]$
$\delta w_n = [0\ \ 0]$
where δw_n represents the displacement of the direct topological neighboring nodes of the winner node in the direction of the input vector ζ.

4. **Selection:** Reset the connections age, if s_1 and s_2 were connected, otherwise create one. Delete all connections with age higher than the predefined maximal age a_{max}. Also, delete nodes that have no emanating connections. Increment the age of all emanating connection of s_1 by one.
   If the nodes s_1 and s_2 are not connected, make a connection between them as below:
   C(s1,s2) = 1
   C(s2,s1) = 1
   The age of connection between s_1 and s_2 is incremented. The value of age starts from zero and updated as below:
   t(s1,s2) = 0
   t(s2,s1) = 0

5. **Recombination:** Every λ iteration, locate the node q with the greatest value of error counter *f*. Create a new node r between q and f with weight $w_r = (w_q + w_f)/2$. Also create connections among r, q and f and delete the original connection between q and *f*. Reduce the error counter of q and f by a multiplication with α. Initialize the error counter of the new neuron with the value of *f*. $w_q = [0.5948\ \ 0.7412]$
   $w_s = [0.9004\ \ 0.1332]$
   $w_r = [(0.5948 - 0.9004)\ \ (0.7412 - 0.1332)]/2 = [0.7476\ \ 0.4372]$
6. **Ageing:** Reduce the error counter of all nodes by multiplication with δ and continue with step one.
7. If a stopping criterion (e.g., net size or some performance measure) is not reached go to step 1.

In this work, we adapted the parameter values used in [39]. The number of nodes fixed by visualizing the clusters generated after

applying GNG on datasets. Although, [81] proposes an algorithm to choose the optimum parameters for GNG. For the optimization of parameters, the authors used the evolutionary algorithm. The future work of this thesis could include such an algorithm to assess the performance of clustering of dataset used. GNG gives as output a certain number of nodes containing data samples. Then, mean and covariance are calculated for each cluster from the subset of samples that belong to each cluster. Nodes inside each GNG can be seen as a set of *letters* containing the main behaviours of GEs. Fig. 4.3a shows an example of GNG clustering of *steering-power* data; the yellow circles represent the nodes and *blue* dots represent the data points associated with them. Moreover, Fig. 4.3b shows an example of one cluster. The *red* cross mark represents the centroid, data points associated to the cluster are shown in *blue* circles and the boundary of cluster is marked by big *red* circle.



(a) Clustering of steering angle and rotor power (S & P).

(b) Example of a cluster.

Figure 4.3: GNG applied on the control dataset (for perimeter monitoring task).

The nodes produced by each GNG is called a set of *letters* containing the main behaviours of generalized errors.

The letters/nodes encoding l-th order derivatives in generalized error for the data combination m (for e.g., S-P) is defined as follows:

$$\boldsymbol{V}^{m,l} = \{V_1, V_2, \ldots, V_n\}, \tag{4.3}$$

where n represents the maximum number of nodes produced by the GNG.

Once performed the clustering operation and obtained the letters belong to each GNG, the next step is to generate the *words*. A *word* can be defined as the coupled nodes activated simultaneously at the same time instance from each GNG. A word can be represented as:

$$W^m = [V_i^0 \;\; V_j^1]^T \tag{4.4}$$

where V_i represents the i^{th} element of the group of nodes produced by GNG1 (which is used to cluster states). Likewise, V_j^1 represents the j^{th} element of the list of nodes produced by GNG2 (i.e., the GNG used to cluster first-order derivatives of states). Then, unique labels assigned for each unique combination of activated couples to form a *dictionary/codebook*.

326x3 double

	1	2	3
1	6	26	84
2	10	3	82
3	20	26	47
4	26	7	154
5	10	26	81
6	6	7	85
7	10	13	142
8	11	3	229
9	6	19	86
10	11	26	228
11	20	3	48
12	26	12	155
13	26	3	202
14	17	3	203
15	10	5	178

Indexes of centroids: GNG1 and GNG2

Unique labels

Figure 4.4: Dictionary (codebook).

The dictionary (list of words) computed for ego-thing *et* for the modality m can be written as:

$$\boldsymbol{D}^m = \{(W^{(1),m}\ L_1), (W^{(2),m}\ L_2), \ldots, (W^{(n),m}\ L_n)\}, \quad (4.5)$$

where L represents each word's unique label, n represents the index of the maximum number of elements in the dictionary.

The example of dictionary is shown in Fig. 4.4, each row represents a word and a unique label assigned to it. The first column consists of the activated nodes belongs, and the activated nodes belong to GNG2 are the elements of the second column of the dictionary.

In this work, we considered the states and it's first order derivatives only, so that the number of GNGs used for the data combination is limited to two. The aim of this step is to learn the discrete variables of the DBN model which is shown inside the yellow shaded area in Fig. 4.6.

– **Estimation of state transition probability**

The letters (see equation 4.3) and the dictionary (list of words) (see equation 4.5) belong to ego-thing *et* have been learned based on the measurements of control variables related to *scenario 1* (refer section 4.2). Based on the timely evolution of *letters* and *words*, transition models can be estimated. Such transition models can expedite the estimation of future *letters* and *words*. The learned *letters* and *words* construct the *superstates* of the DBN model. Once obtained the superstates, the very next step is to estimate probabilistic dependencies between the superstates in DBN (represented as horizontal arrows in yellow shaded area in Fig. 4.6. It is required to make an approximation of temporal probabilistic transition matrix to identify a discrete conditional state transition between nodes for the discrete level of the MJPF. As a consequence, a set of probabilities $p(V^m_{L,k+1}/V^m_{L,k})$(refer Fig. 4.6) have to be estimated, where L is the total number of clusters produced by each GNG.

Transition probabilities between nodes are approximated with rel-

ative frequencies and computed in a transition matrix, which says how much probable the entity goes from the region of a row into a region in a column. The transition matrix is a square matrix whose side is equal to the number of clusters we have found by GNG. To obtain such matrices, the control data sequences in the superstate space of our entity have been considered. Firstly, take one superstate at time instance k and consequent one at time instance $k+1$, then increase of one element in the row of current node and column of next node, that corresponds to absolute frequency. At the end for each row, the sum of elements and divide them by sum, to have the relative frequency of change from superstate V_k to V_{k+1}. Sum of elements of each row must be 1. Afterwards, to see how much time passes from the instant in which object is in one superstate to when it moves to another one, activated superstates over time while executing a certain activity are analyzed to estimate temporal transition matrices encoding the probabilities of staying or passing between the superstates considering time spent in current object's superstate. The probability that entity moves to future superstate can be written in terms of its current superstate and time passed in it as $p(V^i_{k+1}/V^i_k, t)$ where i and j belong to the set of clusters. We observe a series of crossed nodes and compute histograms of elapsed time steps between each couple V_k and V_{k+1}. For each interval of time, there is a different probability to move to a node considering how much time an object spent in a certain node. For each possible transition between superstates, we compute a histogram of how many seconds the object stayed in a node or after how many steps it moved to another one. Then we find the maximum time of all histograms and for each time from 1 to the maximum number of seconds observed, a temporal square matrix whose side is the number of nodes can be computed. For each matrix with corresponding time, it is verified considering histograms. For example, how many times we passed in 5 seconds from node 1 to node 2, and put this value to element $1, 2$ of the matrix corresponding to elapsed time steps $t = 5$. Repeat the procedure for each matrix corresponding to a different number of seconds. After that, we

compute the sum of elements in a row of each matrix and divide row by the sum to obtain an approximation of the probability of passing from the first node to the second one in that number of steps. We can add to our probabilistic model a variable encoding transition between nodes and time elapsed from the previous transition. We call this change of superstate event. It is a discrete asynchronous random variable: an entity can stay in the same superstate, and so we have the null event for many periods, and only certain changes of superstates are possible. Each event is characterized by a label and how much time passed from the event before. An event can be predicted with a bigger time range, and it is a switching variable for superstates as a superstate is a switching variable for continuous states. We can compute the probability of each event knowing the previous event and how much time has passed from the time it happened. For example in this trajectory of discrete variables:
5 5 5 5 5 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 we have two events $a^j_{k1} = (5, 1), a^j_{k2} = (1, 2)$ and time corresponding to second event is $t(a^j_{k2}) = length(1\ 1\ 1\ 1\ 1\ 1\ 1\ 1) - 1 = 8$.
Each event has been given a label depending on the previous node and the following one. Then save sequences of events and corresponding times passed between events. To remove null events, we must not consider cases where an entity remains in the same region and the transitions that have never happened. So looking at the transition matrix's elements, we don't consider elements on the diagonal and null elements and give a label to each not null element. Doing so, we reduce the number of events and the size of the transition matrix of events. It is build incrementing for each event transition from event j to event k element (j,k). Then normalize each line of the matrix to obtain relative frequencies. For each event, corresponding time is stored so that histograms of time elapsed between events can be computed as we did with histograms of transitions between superstates.

An example of the transition matrix is illustrated in Fig. 4.5. If the observation of $s - p$ activates the node 6 of *GNG1*, the transition matrix tells that, 0.35 probability it could belong to

node1, 0.27 probability to node2 and 0.37 to node 8 from *GNG2*. The sum of each row in the matrix will be 1.

10x10 double

	1	2	3	4	5	6	7	8	9	10
1	0	0	0.0517	0.0923	0.2362	0.1808	0	0.0627	0.1292	0.2472
2	0	0	0	0.1579	0.2018	0.2105	0	0	0.1228	0.3070
3	0.3462	0.3590	0.2949	0	0	0	0	0	0	0
4	0	0	0.0665	0.0759	0.2627	0.1076	0.0886	0.0570	0.1361	0.2057
5	0	0	0.1623	0.0755	0.2151	0.0981	0.0868	0	0.1925	0.1698
6	0.3556	0.2741	0	0	0	0	0	0.3704	0	0
7	0	0	0	0.1073	0.2900	0.1469	0	0	0.1751	0.2806
8	0.1310	0.0917	0	0.5328	0	0.1572	0	0	0	0.0873
9	0.3307	0	0.2677	0	0	0.1024	0.1260	0.1732	0	0
10	0.1000	0.1308	0.1231	0	0	0	0.3000	0.3462	0	0

Figure 4.5: State transition matrix.

– **Learned DBN model**

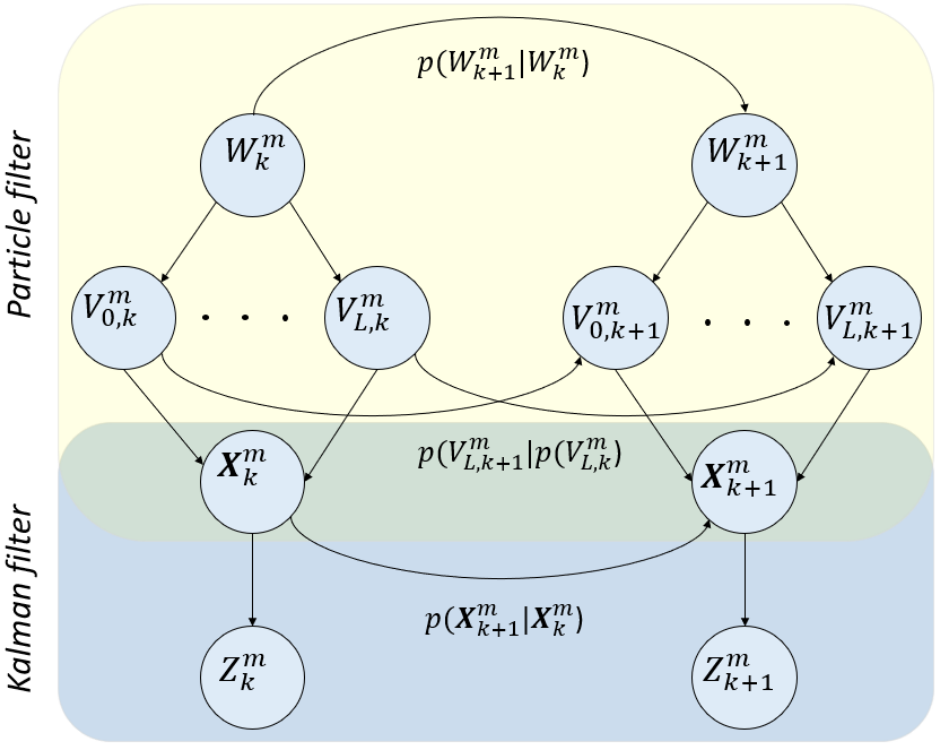


Figure 4.6: Switching DBN model.

All the previous steps are the step by step learning process of the switching DBN model, as shown in Fig. 4.6. The switching DBN model can predict the future states of *ego-thing* in continuous as well as discrete level.

subsectionModel testing and abnormality detection

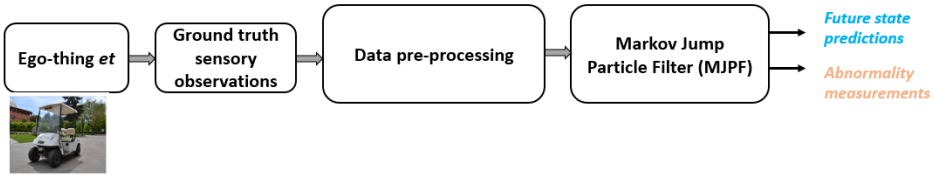


Figure 4.7: Block diagram of online test phase.

Fig. 4.7 shows the block diagram representation of the online test phase. In this phase, we have proposed to apply a dynamic switching model called Markov jump particle filter (MJPF) [14, 16] to make inferences on the DBN models learned in the training phase as shown in Fig. 4.6. The observed sensory data from a new task that the ego-thing has never seen in the offline phase are preprocessed (i.e., normalized) and given as input for MJPF, which computes as output abnormality indicators together with the future state estimation of the Ego-thing.

MJPF is a mixed approach with KF [100] in state space (blue shaded area) and PF [43] in super state space (yellow shaded area) in Fig. 4.6. The MJPF can predict and estimate discrete and continuous states and to detect deviations from the normal model based on anomaly measurements. At each instant, the ego-thing predicts it's own future states by the learned DBN model. By receiving the ground truth observations, the ego-thing can match with the predicted states and detect if any anomalies are present. A detailed description of MJPF is described in Section 3.3 of this thesis and Section II of [14].

Superstates are the total number of clusters obtained by GNG clustering with mean and co-variance associated to each cluster. Superstate space is the space made by the nodes and associated mean and covariances of the the nodes. Such superstates are called switching variables of the DBN model shown in Fig 4.6. Each node is a superstate that encodes observations into discrete components.

In the proposed approach, the posterior probability density function associated with a learned switching DBN model related to modality

m of ego-thing et is :

$$p({W_k}^{et,m}, \tilde{\mathbf{X}}_{\mathbf{k}}^{et,m} / {Z_k}^{(et,m)}) = p(\tilde{\mathbf{X}}_{\mathbf{k}}^{et,m} / {W_k}^{et,m}, {Z_k}^{et,m}) p({W_k}^{et,m} / {Z_k}^{et,m}) \quad (4.6)$$

where ${W_k}^{et,m}$ is the superstate random variable that represents *words* learned through clustering as the higher hierarchical level vocabulary of switching variables; $\tilde{\mathbf{X}}_{\mathbf{k}}^{et,m}$ represents the continuous state of ego-thing et at time instant k.
The MJPF uses Eq. 4.6 to estimate the posterior at the discrete and continuous state. The particles' weight is iteratively computed and allows to approximate the posterior. The predicted particles that have good match with observations get more weight.

- **Abnormality detection and decision making**

  At each time instant, the distance between predicted generalized states and observation evidence of the DBN estimated using probabilistic distances metric called Hellinger distance [62] and an anomlay is detected whenever this distance is higher than a threshold.

  Let $p(\boldsymbol{X}_k^{et,m} | \boldsymbol{X}_{k-1}^{et,m})$ be the predicted generalized states and $p(Z_k^{et,m} | \boldsymbol{X}_k^{et,m})$ be the observation evidence.

  The Hellinger distance can be estimated for modality m of ego-thing et as below:

$$\theta_k^{et,m} = \sqrt{1 - \lambda_k^{et,m}}, \quad (4.7)$$

  where $\lambda_k^{et,m}$ is defined as the Bhattacharyya coefficient [21], such that:

$$\lambda_k^{et,m} = \int \sqrt{p(\boldsymbol{X}_k^{et,m} | \boldsymbol{X}_{k-1}^{et,m}) p(Z_k^{et,m} | \boldsymbol{X}_k^{et,m})} \, \mathrm{d}\boldsymbol{X}_k^{et,m}. \quad (4.8)$$

  The variable $\theta_k^m \in [0, 1]$, where values close to 0 indicate that groungtruth observations match with predictions; whereas values close to 1 shows the presence of an abnormality.

  Example:
  Predicted state, $X_k^{et,m}$ = [-14.5933 5.7686]

Observed ground truth, $Z_k^{et,m}$ = [-13.8000 5.7732]
Bhattacharya distance, d = 0.3147
Bhattacharya coefficient, $\lambda_k^{et,m} = 0.7300$
Hellinger distance = $\sqrt{(1 - \lambda_k^{et,m})} = 0.5196$

4.1.2 Model selection

After learning a set of DBNs, it is possible to select the model that detect abnormalities more accurately. Since each DBN produces a set of abnormality measurements, see equations (4.7) and (4.8), a supervised approach where the testing data is already labelled in abnormal/normal samples; is employed for measuring the performance of the different DBNs when detecting anomalies. Accordingly, the true positive rate (TPR) and false positive rate (FPR) are obtained making it possible to build a set of ROC curves each of them containing the performance of a given DBNs. As is well known, ROC curves plot TPR and FPR at different thresholds, where:

$$TPR = \frac{TP}{TP + FN}\ ; \quad FPR = \frac{FP}{FP + TN} \tag{4.9}$$

In the proposed context, TP is defined as the number of times where abnormalities are correctly identified. FN consists of the times that abnormalities are classified incorrectly. Accordingly, FP are the times where anomalies are wrongly assigned to normal samples and TN represents the times where normal samples are correctly identified .

This work considers two different measurements for selecting the most precise DBN. They are: *(i)* the area under the curve (AUC) of the ROC curves, which quantifies the performance of the DBNs' abnormal detection at several thresholds. *(ii)* The accuracy (ACC) measurement, which is defined as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \tag{4.10}$$

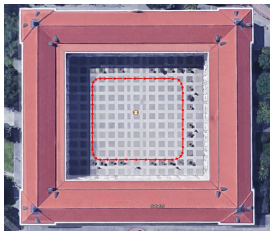
In our context, for each DBN both measurements are obtained and compared for selecting the most precise DBN for abnormality detection

purposes.

4.2 Experimental set up

First of all, it is important to mention that the following experiments were conducted in collaboration with the Intelligent Systems Laboratory, Department of Systems Engineering and Automation of the University Carlos III de Madrid, Spain.

(a) The autonomous vehicle "iCab".



(b) The environment.

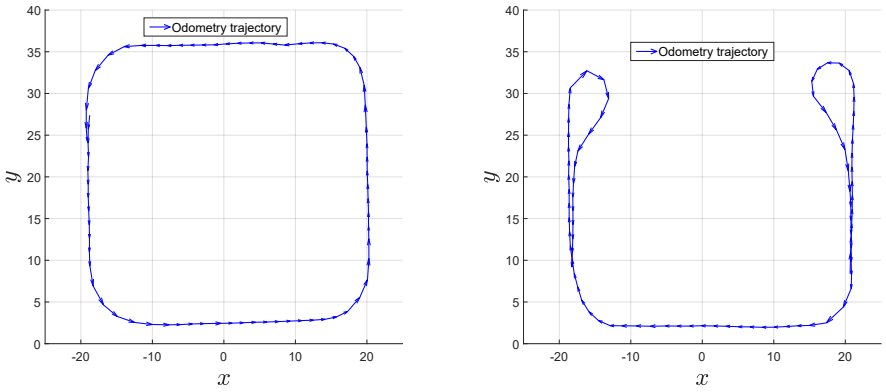
Figure 4.8: The agent and the environment used for the experiments.

4.2.1 Evaluation datasets

We use a dataset collected with the *iCab* vehicle (see Fig. 4.8a), while it moves in a closed environment, see Fig. 4.8b. The dimension of the movement trace in the testing environment is $38m \times 33m$. This chapter considers the internal information of the autonomous system that contains data related to the vehicle's steering angle, rotor velocity, and power. We aim to detect unseen dynamics learned with the proposed method with additional functionality to select the best model. Two following situations are considered:

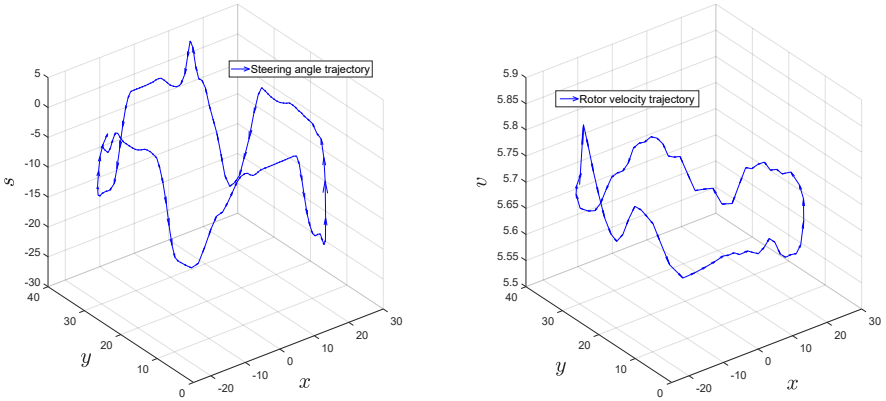
Perimeter monitoring task. The information collected from the perimeter monitoring (PM) experience is employed as training data to learn our models. The vehicle follows a rectangular trajectory along the proposed environment (see Fig. 4.8b). The trajectory data has been plotted and is shown in Fig. 4.9a.

Similarly, the steering angle and velocity information are plotted with respect to position data in Fig. 4.10a and Fig. 4.10b respectively for



(a) Perimeter monitoring. (b) U-turn.

Figure 4.9: Odometry data.



(a) Steering w.r.t position data. (b) Rotor velocity w.r.t position data.

Figure 4.10: Steering/ rotor velocity data in a single lap of PM task.

a single lap of the experience. Additionally, The power information is shown in Fig. 4.11.

U-turn task. The vehicle performs the PM task until it encounters an obstacle. In such a case, the vehicle performs a U-turn maneuver and follows its rectangular path in the opposite direction. The trajectory data of this U-turn manoeuvre is plotted in Fig. 4.9b. The data sequences collected by this experiment are employed as test data to check the model's fitness and detect possible abnormalities.

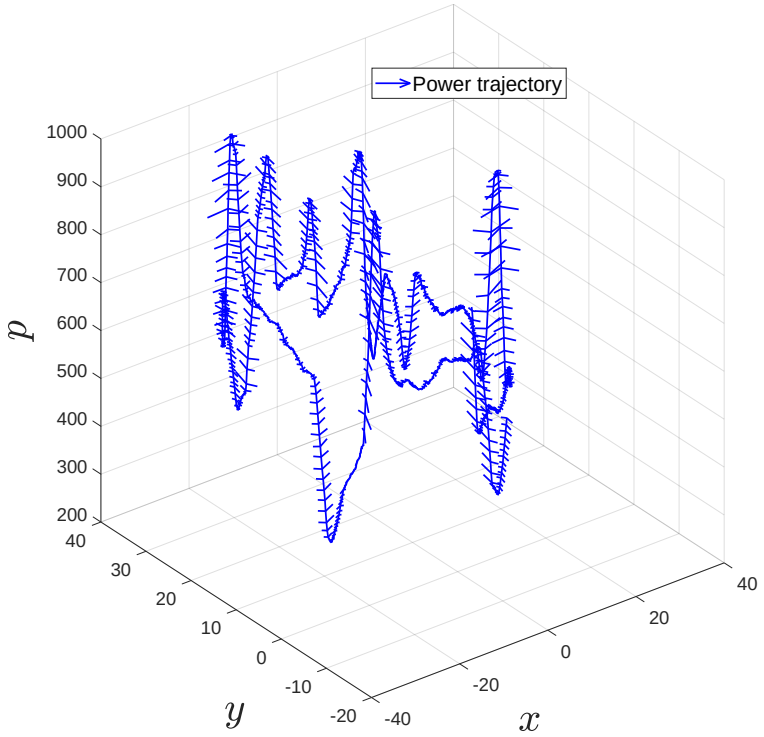


Figure 4.11: Power w.r.t position data in a single lap of PM task.

4.2.2 Set of features for DBN learning

In this work, we use different combinations of the collected information as a set of features from the vehicle. Such features are listed below:

- Steering angle, rotor velocity and power (SVP)
- Steering angle and power (SP)
- Rotor velocity and power (VP)
- Steering angle, rotor velocity (SV)
- Steering angle (S)
- Rotor velocity (V)
- Power (P)

The features above mentioned represent the sensory data (cases) considered for building DBNs. Accordingly, for prediction and abnormality detection purposes we consider seven DBNs that follow the architecture shown in Fig. 4.6. All proposed DBNs are trained based

on the control data from the PM task, see Fig. 4.10 and Fig. 4.11. For state prediction and detection of abnormalities, the control data produced by the U-turn experience is taken into consideration.

4.2.3 Abnormality detection and feature analysis

Based on the DBN feature-cases trained with the PM information, we perform prediction and detection of abnormalities with the features extracted from the U-turn observations. A manual ground truth (GT) of the vehicle's maneuvers is extracted for the U-turn experience. Fig. 4.12 and Fig. 4.13 show the abnormality signals θ_k^m, see equation (4.7), through time for the different DBN feature-cases. Background colors of all plots are the same; they encode the GT of the vehicle's actions in the following way:

- Yellow: Entering U-turn (normal w.r.t PM)
- Violet: U-turn execution (abnormal w.r.t PM)
- Orange: Exiting U-turn (normal w.r.t PM)
- Blue: Inverse curve (abnormal w.r.t PM)
- Green: Straight motion (normal w.r.t PM)

As can be seen, two new maneuvers w.r.t the PM are introduced in the U-Turn task. They are *i)* The U-turn maneuver: which consists of a closed curve not experienced in the PM. *ii)* The inverse curve maneuver: that is present in the lower left/right parts of the trajectory shown in Fig. 4.9b. Such a maneuver was learned in an anticlockwise sense in the PM. However, in the U-turn task, the vehicle performs such a curve also in clockwise direction. Based on the GT explained above, it is possible to build the ROC curve for each different feature-case. Fig. 4.14 contains the ROC curves of the seven feature-cases. It is evident from the plot that SP (curve in blue) performs better than the rest of the features. Table 4.1 summarizes the performance comparison of different features based on the calculation of the two proposed precision measurements, AUC of the ROC and ACC, see equation (4.10). It can be seen that SP presents the highest accuracy in both measurements w.r.t others features with values of 78.20% and

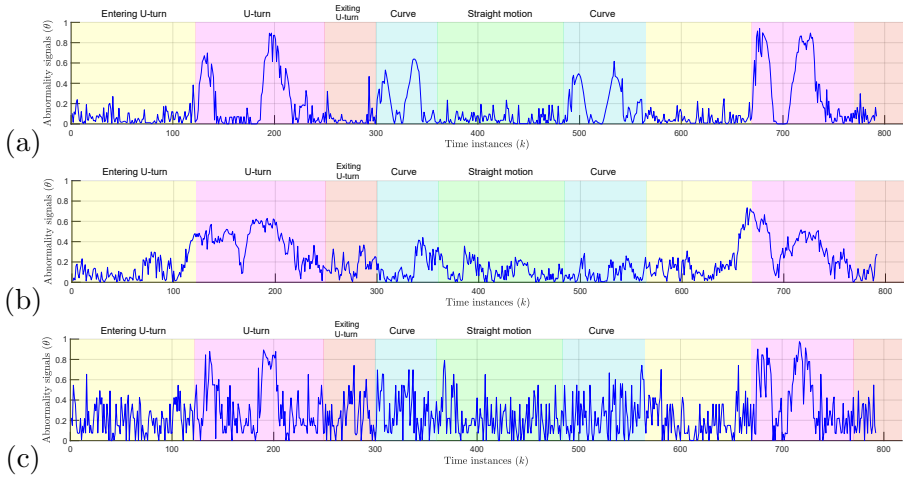


Figure 4.12: Abnormality measurements for single variable features: (a) S, (b) V, and (c) P

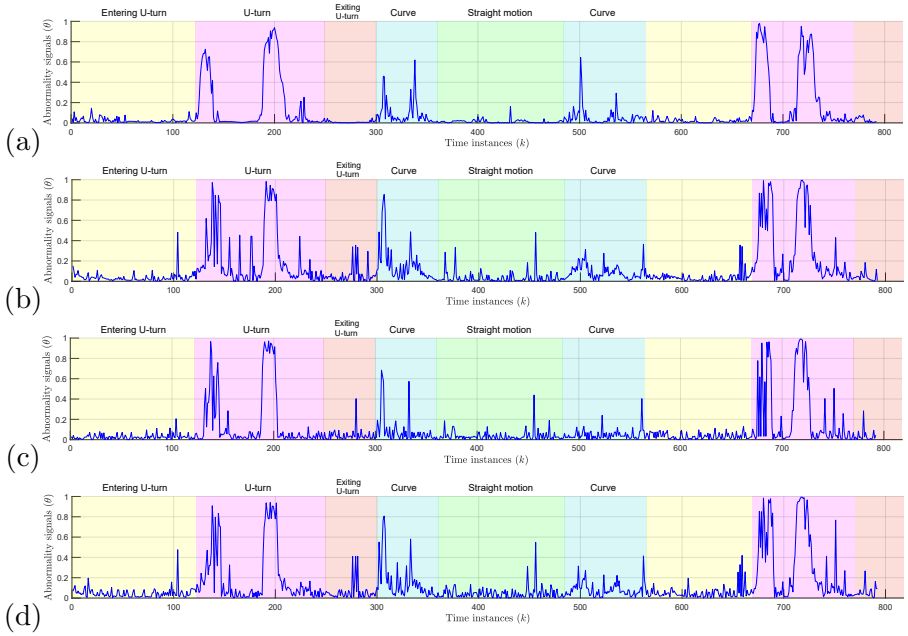


Figure 4.13: Abnormality measurements for mixed variable features: (a) SV, (b) SP, (c) VP and (d) SVP

76.11% respectively. Such results suggest that the DBN trained with the SP information provides the best recognition of abnormalities and the most accurate predictions of future instances.

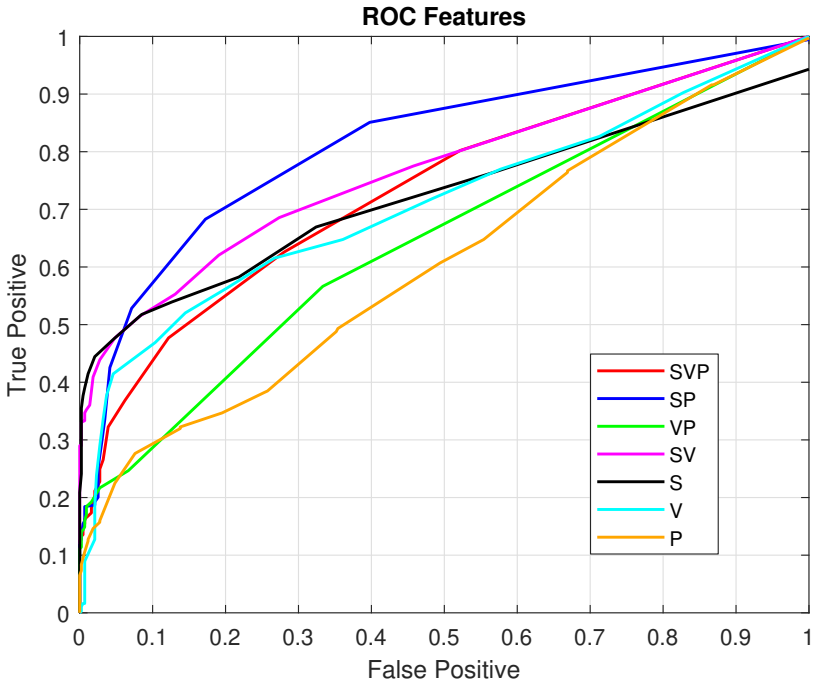


Figure 4.14: ROC for different features.

Feature	SVP	SP	VP	SV	S	V	P
AUC (%)	72.99	**78.20**	64.27	76.76	71.44	70.71	60.44
ACC (%)	69.40	**76.11**	62.56	73.38	73.38	70.64	62.68

Table 4.1: Precision measurements.

4.3 Chapter summary

This chapter proposes and tests a method for selecting the most precise DBN when predicting abnormalities in real scenarios where multiple sensory data is analyzed. The method is evaluated with real data collected from a moving vehicle performing different tasks in a closed environment. Moreover, to evaluate and compare the model performance, the ROC curve is plotted. The area under the curve (AUC) and accuracy (ACC) helps to select the best model to predict agents' future states and detect the abnormality.

Results suggest that the proposed method recognizes the features from a set of sensory data that facilitates identifying previously unseen maneuvers, i.e., abnormal situations. Thus, the proposed method can be

useful for selecting relevant features when dealing with many features in networking operations.

Chapter 5

Implementing Collective Awareness in a network of Ego-things: Phase I

The advancements in connected and autonomous vehicles in these times demand the availability of tools providing the agents with the capability to be aware and predict its own states and context dynamics. This chapter presents a novel approach to develop an initial level of collective awareness (CA) in a network of intelligent agents. A specific collective self-awareness functionality is considered, namely agent-centred detection of abnormal situations present in the environment around any *agent* in the network. Moreover, the agent should be capable of analyzing how such abnormalities can influence the future actions of each *agent*. Data-driven dynamic Bayesian network (DBN) models learned from time series of sensory data recorded during the realization of tasks (agent network experiences) is here used for abnormality detection and prediction. A set of DBNs, each related to an *agent* is used to allow the agents in the network to reach synchronously aware about possible abnormalities occurring when available models are used on a new instance of the task for which DBNs have been learned.

A growing neural gas (GNG) algorithm is used to learn the nodes variables and conditional probabilities linking nodes in the DBN models;

a Markov jump particle filter (MJPF) is employed for state estimation and abnormality detection in each agent using learned DBNs as filter parameters. Performance metrics are discussed to asses the algorithm's reliability and accuracy. The impact is also evaluated by the communication channel used by the network to share data sensed in a distributed way by each agent of the network. The IEEE 802.11p protocol standard has been considered for communication among agents. Performances of the DBN based abnormality detection models under different channel and source conditions are discussed. The effects of distances among agents and of the delays and packet losses are analyzed in different scenario categories (urban, suburban, rural). Real datasets are used acquired by autonomous vehicles performing different tasks in a controlled environment.

In the previous chapter (Ch: 4), we have considered only one agent to develop multi-modal self-awareness to detect abnormal situations. A Hellinger distance metric is used to estimate the abnormality. The ROC curve is plotted to evaluate and compare the different models' performance developed from various exteroceptive and proprioceptive sensory data. The area under the curve (AUC) and accuracy (ACC) helps to select the best model to predict agents' future states and detect the abnormality.

This chapter focuses on developing collective awareness in an agent network to detect collective abnormalities. Here, we have considered only a single modality and different channel conditions and protocols. As in the previous chapter, the Hellinger distance metric is used to estimate the abnormality. However, in this chapter, the ROC curve is used to compare the performance of the models under different channel conditions and data rates. The area under the curve (AUC) and accuracy (ACC) are the main metrics used to evaluate the model's performance to know the performance is under acceptable range or not.

5.1 Collective awareness modelling

The collective awareness in a network of ego-things is defined here as the capability of a set of ego-things in a network to understand whether perception-action information processing models they are provided are performing normally. Normality is defined in a Bayesian inference sense, i.e., as the capability of dynamic models describing hidden object state characteristics as confirmed by observations of available agents ego-things' sensors. Such an ability is provided to each ego-thing in the network and concerns the whole set of agents. Communication is available in the network to exchange information necessary to detect abnormalities of all ego-things by each agent in the network. Each ego-thing can so achieve awareness not only about the fitness of its own models when predicting its own state but also about the possibility that abnormality conditions affect the actions of other cooperating agents with respect to predictions provided by their dynamic models. Such a collective awareness can trigger agents' decision systems to perform emergency routines or switching to other available modalities.

The collective awareness is based on detecting jointly and synchronously abnormal situations present in the context. It allows appropriate decisions can be taken to maintain the stability of the entire network of systems.

The ego-things are equipped with various sensors. The collected data from each ego-thing have been initially synchronized and then categorized into different groups. In this work, we mainly consider the data related to the control part of the ego-things and the trajectory data to develop collective awareness. The proposed method is divided into two parts: offline training and online testing. A block diagram representation of the proposed method where offline training and online processing carried on by each ego-thing in the network is shown in Fig. 5.1. During the offline training phase, each ego-thing learns probabilistic filtering models from agent sensors' dynamic data series collected while collectively performing a reference situation task. This implies that all agents perform the task autonomously or in a teleoperated way during the training phase. The collected data series provides

information of data collected by sensors' observing exteroceptive and proprioceptive data. By assuming that observation models can remain invariant along the process (i.e., the model for estimating state likelihood from sensory observations is given and fixed), a set of dynamic models is learned, composed by a discrete vocabulary of continuous conditional probabilities functions and by a transition matrix. Such models are organized within a DBN. Such a process is repeated for each agent, and the set of DBNs related to each agent is made available to the collective ensemble of ego-things. In the online phase, each DBN is used for filtering agent sensory data within each agent. The comparison of learned dynamic prediction models with incoming observations allows each agent to estimate the level of fitness and to measure abnormality of the collective situation in a distributed way. However, to this end, communications have to be maintained to allow each agent to filter and detect abnormalities also of other ego-things in the network. Filtering is performed using a Bayesian filter appropriate for the type of DBNs learned, i.e., switching models. Markov jump particle filter (MJPF) has been here chosen as the dynamic probability models in learned DBNs are here linear and continuous Gaussian, so allowing Kalman filters to be used at a continuous level in switching models. In Fig. 5.1, it is highlighted how such filters are here provided with the additional capability of measuring abnormalities in addition to filtering, and such capability is at the basis of CA.

5.1.1 Model learning phase: offline

Each ego-thing will learn a switching DBN model for itself in the training phase, i.e., from the data collected by its own sensors, and one DBN model for each other ego-thing present in the network by exploiting the data generated by that ego-thing's sensors. In this work, ego-things are autonomous vehicles, and the number of vehicles is limited to two. In Fig. 5.1, the first part (gray shaded area) represents the training phase of an ego-thing, and the steps followed to learn the switching DBN models are explained below.

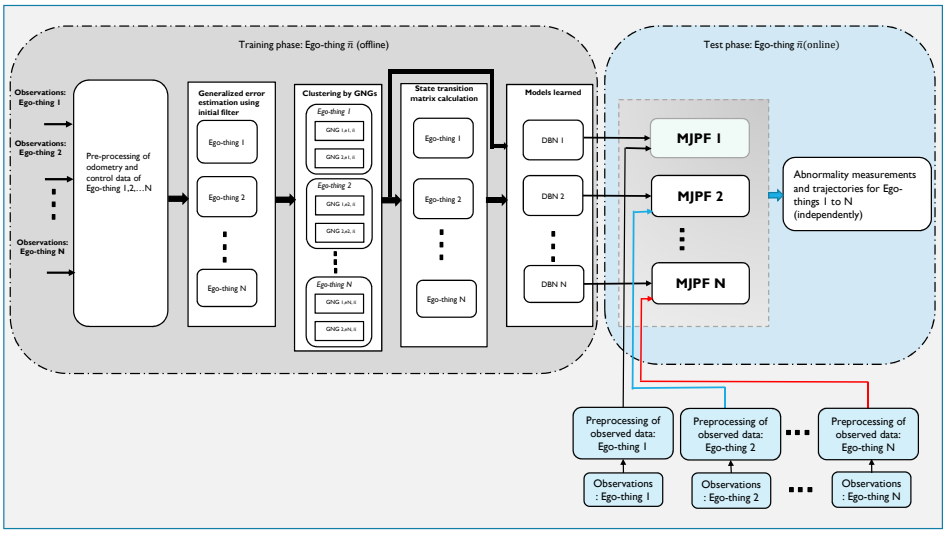


Figure 5.1: Block diagram: training phase and test phase.

Data preprocessing and state estimation

First, the collected multisensory data are synchronized by using their time stamps. In this work, we considered as a case study the data sequences related to two low dimensionality sensorial data, namely odometry as representative of ego-thing's exteroceptive sense of position and steering as proprioceptive control information of the ego-thing. An initial basic generalized filter [27] has been applied to the data sequence for the estimation of generalized errors (GEs). The generalized error estimation of acquired data is described below.
Let $Z_k^{e\overline{n}}$ be the measurements in the ego-thing $e\overline{n}$ at the time instant k and $X_k^{e\overline{n}}$ be the state associated to the measurement $Z_k^{e\overline{n}}$, such that $Z_k^{e\overline{n}} = g(X_k^{e\overline{n}}) + \omega_k$. $g(\cdot)$ is the function that maps states into observations and ω_k represents the noise of the sensors. Similarly, $e\overline{n}$ will also have measurements from all the other ego-things in the network, which can be represented as $Z_k^{e1}, \ldots, Z_k^{en}, \ldots, Z_k^{eN}, n \in \mathcal{N}, n \neq \overline{n}$, where N and $\mathcal{N}$ are the number and the set of ego-things in the network, respectively.

As explained in [37, 10], including time derivatives in hidden object states allows dynamic probabilistic flow models describing ego-thing states to be one to one related with descriptors of motion laws coming from the mechanical statistic, i.e., Lagrangian. Moreover, such flow models are represented in the moving reference system of each ego-thing, allowing data series to be described as relative not only to the estimated state of the object but also to how such a state is instantaneously changing. The generalized error (GE) of $e\overline{n}$ by considering only itself can be defined as:

$$\boldsymbol{X}_k^{e\overline{n}} = [X_k^{e\overline{n}} \;\; \dot{X}_k^{e\overline{n}} \;\; \ddot{X}_k^{e\overline{n}} \cdots X_k^{(L)e\overline{n}}]^\intercal, \tag{5.1}$$

where (L) indexes the L-th time derivative of the state.

The l-th time derivative in $e\overline{n}$ at the time k by considering only itself can be approximated as:

$$X_k^{(l)e\overline{n}} = \frac{X_k^{(l-1)e\overline{n}} - X_{k-1}^{(l-1)e\overline{n}}}{\Delta k}, \tag{5.2}$$

where $X_k^{(0)e\overline{n}} = X_k^{e\overline{n}}$ and Δk is the uniform sampling time for all multisensory data.

The generalized error of $e\overline{n}$ by considering all the ego-things in the network can be written as:

$$\boldsymbol{C}_k^{e\overline{n}} = [\boldsymbol{X}_k^{e1} \;\; \boldsymbol{X}_k^{e2} \;\; \boldsymbol{X}_k^{e3} \cdots \boldsymbol{X}_k^{eN}]^\intercal, \tag{5.3}$$

Clustering by GNG algorithm

When a data series is available, a generative filter capable of generating other instances provided of the same statistical properties as well as predicting future states has to be learned. Generative filters here used are hierarchical switching 2-Time Slice DBNs (2T-DBN) [58]. This generative filter, as shown in Fig. 5.2 is composed of hidden states at continuous and discrete levels. Generalized error are here used at the continuous level. Discrete hidden states are hierarchically higher

and represent switching variables. For each value of such random variables, a different dynamic model has to be learned at the continuous level capable of predicting in a different way dynamics of states. This type of DBNs is capable of representing non-linear dynamic models by using a set of linear dynamic models. In order to learn such DBNs, the vocabulary of switching variables and the associated set of dynamic linear models must be learned from data. To this end, having used generalized error is particularly useful. In fact, a technique can be used as in [27] that allows defining an initial basic generalized filter [37] that operates on data series to produce an estimation of the dynamic model that should be associated to each sparse state point obtained by filtering the data sequence. Such technique consists of an initial filter based on a single value switching variable; such value corresponds to a unique dynamic model that assumes no state change should be associated with values in the data series. When a data series violates this assumption, obtained derivatives of state correspond to errors with respect to such a hypothesis. Errors can be clustered to define a set of state-dependent linear dynamic models characterizing the state as varying according to average derivatives and their covariances. Jointly clustering in an unsupervised way, states and errors allow one to obtain a vocabulary of regions. Each region is characterized by a compact part of the state space and by a compact subspace of the derivative state space. The average state derivative in the region subspace defines a different filter for each compact state subspace. Here we used an unsupervised clustering approach, the Growing Neural Gas (GNG) [39] method to obtain regions from generalized errors produced as outputs by the initial filter, i.e., sequences of coupled state estimations and errors. GNG clusters correspond to coupled compact regions of state points and errors. Derivative errors cluster encode the description of the expected dynamics that caused the data series to vary instead of following the hypothesis of the initial filter. Different ways of changing are coded as behaviors that have been found in a corresponding compact state region. The compact state region represents switching variables in the hierarchical DBN. The reason why we choose GNG algorithm for clustering, pseudo code with example is described in Section 4.1.1.

The output of GNG consists of a set of clusters defined as nodes. A separate GNG clustering is applied to states and derivatives obtained from the initial filter in the proposed approach. Each node groups a subset of samples of states or derivatives with a low distance to the center of mass of the region associated with the node. Iterative presentation of the same set of samples allows reorganization of nodes averages until convergence is reached. Nodes produced by GNG can be seen as a set of *letters* forming a vocabulary. A different vocabulary is formed for GNGs working on state and derivative samples produced by the initial filter. Nodes associated with the GNG working on derivatives form a vocabulary of dynamic linear models. The flow model of each dynamic model is defined by the center of mass of the error in the respective node. On the other side, nodes associated with the GNG working on states define a vocabulary of regions, i.e., switching variables of the state space. The set of nodes produced as output by GNG l, i.e., related to the l-th derivative order, of the ego-thing $e\overline{n}$ can be defined as:

$$\boldsymbol{V}^{(l)e\overline{n}} = \{V_1^{(l)e\overline{n}}, V_2^{(l)e\overline{n}}, \ldots, V_{G^{(l)e\overline{n}}}^{(l)e\overline{n}}\}, \tag{5.4}$$

where $G^{(l)e\overline{n}}$ is the set of nodes of the GNG l related to ego-thing $e\overline{n}$'s l-th derivative of the state.

$V_n^{(l)e\overline{n}}$ defines the node, and it is considered as a Gaussian random variable whose mean value is the average of samples and whose size corresponds to the variance of the samples themselves. $\boldsymbol{V}^{(l)e\overline{n}}$ can be seen as a *vocabulary* of order l composed by the relative nodes. The switching variables at the highest level of the DBN model learned by GNGs are so computed in Fig. 5.2 is the switching variable. Such variable assumes values from the vocabulary learned by GNG working at the state level $l = 0$. Each region can so be seen as a switching variable: Each value of the variable indexes a region in the continuous state space corresponding to a Gaussian having as mean and covariance associated with the node. The dynamic model associated with that region is found by identifying a letter in the vocabulary of higher derivatives GNG nodes that specifies the velocity and higher-order generalized coordinates of a set of dynamic models that can be

associated with the state region.

The compact regions of the derivative state space form a *vocabulary* composed by symbols associated with different dynamics of generalized error $\boldsymbol{X}_k^{e\overline{n}}$. In this work, generalized errors include only states and their first order derivatives such as $\boldsymbol{X}_k^{e\overline{n}} = [X_k^{e\overline{n}} \;\; \dot{X}_k^{e\overline{n}}]^\intercal$. A generic element (letter) of the vocabulary describing clusters of state derivatives can be associated with an equation of a dynamic model to be used by a linear filter. Such a model can be written as:

$$\boldsymbol{X}_{k+1}^{e\overline{n}} = A\boldsymbol{X}_k^{e\overline{n}} + BU_k + w_k \tag{5.5}$$

where

$$A = \begin{bmatrix} I_j & 0_{j,j} \\ 0_{j,j} & 0_{j,j} \end{bmatrix} \; ; \; B = \begin{bmatrix} 0_{j,j} \\ I_j \Delta k \end{bmatrix}$$

The variable j is related to the dimensionality of the state vector for data under consideration. I_j is an identity matrix of dimension j. $0_{j,j}$ is a zero $j \times j$ matrix. $w_k \sim \mathcal{N}(0, \sigma)$, encodes the noise produced by the system. U_k is a control vector that is defined from the average derivative of states obtained by GNG within a dynamic model region, The dynamic model to be chosen is one of the state regions to which $\boldsymbol{X}_k^{e\overline{n}}$ belongs. A different dynamic model can be associated with different letters describing the same state space region.

By combining letters of nodes produced by GNG working on different derivatives, it is possible to obtain a set of *words* which define discrete states combined with dynamic models, so providing a semantic vocabulary whose elements combine centroids of different derivative orders. A word computed at ego-thing $e\overline{n}$ is defined as:

$$W^{e\overline{n}} = [V_i^0 \;\; V_j^1]^T \tag{5.6}$$

where V_i^0 represents the i^{th} element of the group of nodes produced by GNG1 (which is used to cluster states). Likewise, V_j^1 represents the j^{th} element of the list of nodes produced by GNG2 (i.e., the GNG used to cluster first-order derivatives of states). A unique label is assigned to each coupled nodes and formed a dictionary and can be written as:

$$\boldsymbol{D}^{e\overline{n}} = \{(W^{(1),e\overline{n}}\ L_1), (W^{(2),e\overline{n}}\ L_2), \ldots, (W^{(n),e\overline{n}}\ L_n)\}, \qquad (5.7)$$

where L represents each word's unique label, n represents the index of the maximum number of elements in the dictionary.

The switching variable acts as a variable at a higher hierarchical level that explains the states from a semantic viewpoint. The discrete switching variables (i.e., *letters* and *words*) of the learned DBN model shown in the pink shaded area in Fig. 5.2.

Estimation of state transition

The vocabularies are learned by applying initial filters and GNG clustering to each ego-thing $e\overline{n}$ sensory data acquired along a cooperative task performed with other ego-things. For example, in scenario 1 (refer section 5.2) a cooperative driving task of two autonomous cars is considered. In order to allow each ego-thing to develop models that consider time evolution not only at continuous level but also as probabilistic transitions among *words* in the learned *vocabularies*, timestamps are assumed to be provided to data series, and transition models to be used at the discrete level of DBNs are estimated. Such transition models allow switching variables to be predicted probabilistically at each moment by the DBN. Moreover, as the DBNs estimate at each time a joint posterior probability over switching models and continuous states, the predictions provided by the transition model can be used as a source to obtain a further measurement of semantic abnormality. In particular, if predicted words do not match with evidence supported by observations of one agent, then such an agent can occur in a semantic abnormality.
The probabilistic transition matrix has been estimated from the data sequence by considering the transitions in time, and such matrix can tell the mapping of the variables in discrete space (i.e., word space). In other words, it can tell the probability of transition from word W_k^{en} at time instance k to the word W_{k+1}^{en} in next time instance $k+1$ shown in Fig. 5.2. We use this information for the prediction purpose in the word level.

DBN model for all the agents

All the previous steps are the step-by-step learning process of the switching DBN models. Each ego-thing learns a total number of N switching DBN models to predict the future states of each entity in continuous and discrete levels. The set of DBNs learned by each ego-thing ei and ej is the same for each other ego-thing in the network and can be written as:

$$\boldsymbol{DBN}^{ei} = \{DBN^{e1}, \cdots, DBN^{eN}\} = \boldsymbol{DBN}^{ej}, \forall i, j \in \mathcal{N} \tag{5.8}$$

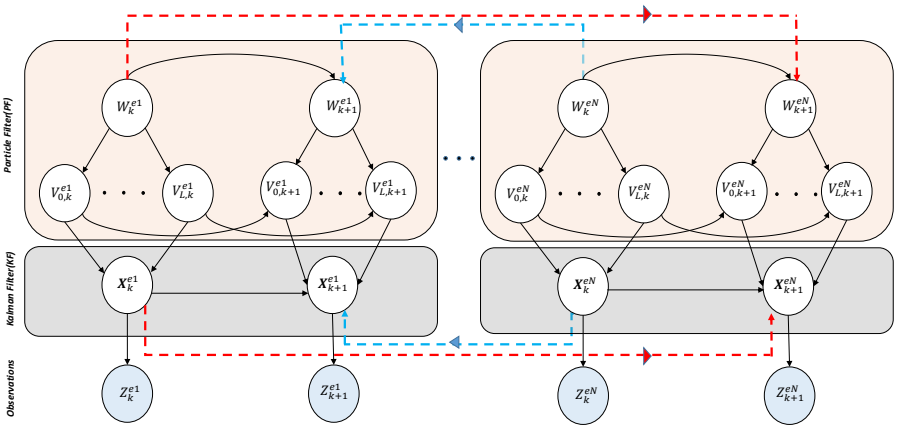


Figure 5.2: CDBN model.

The number of DBNs learned can be represented as shown in the Fig. 5.2. In each DBN, there are three levels such as measurements, continuous and discrete levels. The lowest level of the model is the measurement/observation level, and *blue* nodes represent the variables. The *gray* shaded area belongs to the continuous level, and the *orange* shaded areas represent the discrete level of the model. The arrows and links (coloured in black) of such DBN are learned based the *Scenario 1* task (see section 5.2). The horizontal arrows in black represent the conditional probability between two consecutive time instances at continuous and discrete levels. The vertical arrows describe the causalities between inferences of ego-things at discrete, continuous states and

observation levels. Moreover, the red and blue dotted arrows represent the influence of one ego-thing's action on the future states of the other ego-things in the network. The dotted arrows represent how one ego-things action can be influenced by the future actions of the other ego-things in the network and vice versa.

5.1.2 Online state prediction and anomaly detection

In Fig. 5.1, the second part (shaded in blue) shows the block diagram representation of the online test phase. In this phase, we have proposed to apply a dynamic switching model called MJPF [14, 51] to make inferences on the DBN models learned in the training phase as shown in Fig. 5.2. MJPF is a Bayesian filter with a Kalman filter (KF) is associated with each particle. In MJPF we use Kalman filter (KF) [100] in state space (grey shaded area) and particle filter (PF) [43] in higher hierarchical level called *word* level (pink shaded area) in Fig. 5.2. The blue and red arrows in Fig. 5.2 depict the information exchange between two ego-things, and, as a consequence, two DBN models. Those arrows tell how the future states of one ego-thing can influence the next states of the other one.

Estimation of future states

The MJPF is able to predict and estimate discrete and continuous states of the ego-things. In addition to that, it produces another information, i.e., abnormality measurements.
The data sequence (experience) never seen in the online training step is pre-processed and given as input to the MJPF applied on learned DBN models. The output of each MJPF is the estimation of future states of the associated ego-thing along with the probabilistic and spatial abnormality measurements. A detailed description of MJPF is described in Section 3.3 of this thesis and Section II of [14].

MJPF uses PF for discrete variables, here corresponding to word variables; each particle used to approximate the joint posterior in MJPF is augmented with an associated continuous random variable characterized by a Gaussian probability. In our case, the dynamic model

describing changes of the continuous variable associated with a given word value is represented as in Eq. 5.5, while the transition model is used as a dynamic model at the word level. In the prediction step of the MJPF, a SIR [105] PF approach is used to predict new candidate particles at the next step using the transition model at the discrete level. Each particle is also enriched by a Gaussian prediction of the continuous associated variable. This is done as in Kalman filter, being dynamic models and observation models are linear, and variables are Gaussian. Each predicted particle is so characterized as a word of given valued with an associated prior probability at a continuous level. In the update step, the ground truth observations (belonging to each ego-thing) are used to first update the prior at continuous level, so obtaining the new posterior, and then providing a new weight to the particle word, based on the evidence that such a posterior provides to the specific word itself. In our approach, these traditional MJPF are enriched by the computation of abnormality measurements as described in the next section to allow agents to be aware of the fitness of their dynamic models to the observed sequences.

The posterior probability density function of MJPF belongs to ego-thing $e\overline{n}$ can be written as:

$$p({W_k}^{e\overline{n}}, {\boldsymbol{X_k}}^{e\overline{n}}/{Z_k}^{e\overline{n}}) = p({\boldsymbol{X_k}}^{e\overline{n}}/{W_k}^{e\overline{n}}, {Z_k}^{e\overline{n}}) p({W_k}^{e\overline{n}}/{Z_k}^{e\overline{n}}) \quad (5.9)$$

where ${W_k}^{e\overline{n}}$ is the word in the higher hierarchical level and ${\boldsymbol{X_k}}^{e\overline{n}}$ is the continuous state in the state space belongs to ego-thing $e\overline{n}$ at time instant k .
As stated above, a different Kalman filter is associated with each particle ${W_k}^*$ and is different for each discrete zone (cluster). The Eq. 5.9 shows the link between the discrete state (i.e. words) and continuous state estimation. The KF associated to particle ${W_k}^*$ is used to estimate the prediction on the continuous state ${\boldsymbol{X_k}}^{e\overline{n}}$ and to estimate $p({\boldsymbol{X_k}}^{e\overline{n}}/{W_k}^{e\overline{n}}, {Z_k}^{e\overline{n}})$.

As explained before, each ego-thing has its own switching model as well as the model of other ego-things. At each instant, the ego-thing predicts its own future states and future states of the other ego-things

by the learned switching DBN models. By receiving the ground truth observations, the ego-thing can match with the predicted states and detect if any anomalies present. The observations from other ego-things can be received through the established wireless channel with a certain delay and loss. By making efficient communication between the ego-things, we can develop collective awareness in the entire network of ego-things. Such collective awareness can tell if any abnormal situations happen anywhere in the network. Moreover, the collective DBN models can handle the uncertainty of the environment and the variability of observations.

Abnormality detection

Posterior probability estimation in MJPF is here enriched with computation of additional information useful for self-awareness of individual ego-things. i.e., abnormality measurements. Such information is estimated to instantaneously allow each ego-thing to measure how well the learned models fit the currently observed sequence. To estimate the abnormality of a sequence, a statistical distance metric is here proposed that estimates the distance between predictions performed within MJPF at discrete and continuous levels and the sensory observations produced along with an ego-things experience. In this work Hellinger distance (HD) [19] is proposed as the metric to evaluate sequence abnormality.

Some important statistical distances include Bhattacharya distance [21], Hellinger distance [19], total variation distance [98], etc. The Bhattacharyya distance measures the similarity of two probability distributions. It is closely related to the Bhattacharyya coefficient, which is a measure of the amount of overlap between two statistical samples or populations. Similarly, the Hellinger distance (closely related to, although different from, the Bhattacharyya distance) is used to quantify the similarity between two probability distributions. The Hellinger distance is defined between vectors having only positive or zero elements [2]. The datasets used in this chapter are normalized, so the values vary between zero and one; there aren't any negative values. For this reason, Hellinger distance is more appropriate than

using other distance metrics as abnormality measures. The works in [51] and [62] used Hellinger distance as an abnormality measurement.

In this work, Hellinger distance is used as an abnormality measurement between predicted generalized errors and observation evidence.

The Hellinger distance related to the ego-thing $e\overline{n}$ can be written as:

$$\theta_k^{e\overline{n}} = \sqrt{1 - \lambda_k^{e\overline{n}}}, \tag{5.10}$$

where $\lambda_k^{e\overline{n}}$ is defined as the Bhattacharyya coefficient [21], such that:

$$\lambda_k^{e\overline{n}} = \int \sqrt{p(\boldsymbol{X}_k^{e\overline{n}} | \boldsymbol{X}_{k-1}^{e\overline{n}}) p(Z_k^{e\overline{n}} | \boldsymbol{X}_k^{e\overline{n}})} \, \mathrm{d}\boldsymbol{X}_k^{e\overline{n}}. \tag{5.11}$$

The variable $\theta_k^m \in [0, 1]$, where values close to 0 indicate that groungtruth observations match with predictions; whereas values close to 1 shows the presence of an abnormality.

Once detected abnormal situations, the ego-thing has to take appropriate actions either by stopping itself or reducing the speed, etc. However, the decision-making part is not included in this work.

5.1.3 Evaluating the model performance after the packet loss

Each ego-thing has its own model for prediction of the future states and the ground truth observations received from the sensors to check whether if any anomalies present in the environment around it. At the same time, the models for other ego-things can predict the future states and receive ground truth observations from the corresponding ego-things with a certain delay and loss in transmission. The DBN model of each agent is updated sequentially (with a certain delay) using the shared information. The delay and the loss depend on various factors such as the distance between the ego-things, the employed communication protocol, modulation scheme, and frequency, scenario conditions (urban, rural, ...), etc.

To check the model performance in predicting abnormal situations the true positive rate (TPR) and false positive rate (FPR) are calculated

to build a set of ROC curves [50] corresponds to different K factor values [108]. The ROC curves plot TPR and FPR at different thresholds, where:

$$TPR = \frac{TP}{TP + FN}\ ; \quad FPR = \frac{FP}{FP + TN} \tag{5.12}$$

The true positive (TP) is defined as the number of times where abnormalities are correctly identified. False-negative (FN) consists of the times that abnormalities are classified incorrectly. Accordingly, false positive (FP) are the times where anomalies are wrongly assigned to normal samples, and true negative (TN) represents the times where normal samples are correctly identified. In this work, mainly considered two parameters of ROC for measuring the performance of the model before and after the transmission loss are: *(i)* the area under the curve (AUC) of the ROC curves, which quantifies the performance of the DBNs' abnormal detection at several thresholds; *(ii)* the accuracy (ACC) measurement, which is defined as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \tag{5.13}$$

Communications among ego-things

In the training phase, we assumed that each ego-thing has available all the required data describing all the other ego-things' status. However, in a real scenario, data exchange among ego-things through a wireless mean has to be considered. Different variables affect communication performance over time. They are mainly related to:

- objects' movement, such as object's velocity, acceleration, and moving direction;
- environment where the objects are located, such as urban or rural scenario, presence of obstacles, Line of Sight (LoS) or Non-LoS (NLoS) conditions;
- chosen communication parameters, such as employed communication protocol and modulation, exploited frequency band, achievable data rate, transmission power, and received signal strength.

The channel among ego-things has to be properly modeled to consider all the effects that can affect the obtained performance, such as scattering, diffraction, reflection, shadowing, and fading.

The effects on the wireless channel are addressed by large-scale and small-scale channel models. Large scale models cover effects such as path loss and the effects of the propagation environment over large distances. Small scale models, on the contrary, describe the behavior in the time domain, taking into account the fast fading effects, i.e., multipath propagation. Large and small-scale models are combined to shape channel behaviors realistically. Received power P_r is composed of the transmit power P_t, the large scale effects, i.e., path loss P_L, and the small scale effects ζ:

$$P_r = P_t P_L \zeta \tag{5.14}$$

The path loss is the radio attenuation due to the communication mean. It is mainly affected by the communication frequency f and the distance d between source and destination. It can be computed as:

$$P_L = \left(\frac{\lambda^2}{(4\pi)^2 d^\alpha} \right) G_R G_T \tag{5.15}$$

where $\lambda = 2\pi f$, α is the attenuation factor, G_R and G_T are the reception and transmission antenna gains, respectively.

The presence of objects and obstacles in the environment originates multiple copies of each transmitted signal, which can strengthen (if $\zeta > 1$) or weaken (if $\zeta < 1$) the original signal. This effect is called multipath fading and can be modeled as a Rayleigh, Rician, or Nakagami distribution.

Considering the current state-of-the-art, we focus on a Rician channel model based on a Rice distribution when LoS is present. Rice distribution can be expressed with parameters K and P_r, which are the Rician K factor and the received power, respectively, or as a function of ρ and σ, which are field strength of the LoS component and the field strength of scattered components, respectively.

The Rice distribution is:

$$p_Z(z) = \frac{z}{\sigma^2} exp\left(\frac{-z^2-\rho^2}{2\sigma^2}\right) I_0\left(\frac{z\rho}{\sigma^2}\right) \tag{5.16}$$

where $z \geq 0$, ρ and σ are the signal strength of the dominant and of the scattered paths, respectively. Therefore, ρ^2 and $2\sigma^2$ are the average power of the LoS and NLoS multipath components, respectively. As the direct wave weakens, the Rice distribution becomes Rayleigh.

Rician K factor is defines as :

$$K = \frac{\rho^2}{2\sigma_0^2} \tag{5.17}$$

It expresses the ratio between the dominant component to scattered waves. The stronger the line of sight component, the greater the K factor. In this way, the Rice distribution in eq. (5.16) can be expressed in terms of linear K factor as:

$$\begin{aligned} p_Z(z) =& \frac{2z(K+1)}{P_r} exp\left(-K - \frac{(K+1)z^2}{P_r}\right) \cdot \\ & \cdot I_0\left(2z\sqrt{\frac{K(K+1)}{P_r}}\right) \end{aligned} \tag{5.18}$$

where I_0 is the modified Bessel function of first kind and zero order [41]. When $K \to \infty$, the Rice distribution tends to a Gaussian one, and when $K \to 0$, i.e. in case no dominant direct path exists ($\rho = 0$), the Rician fading reduces to a Rayleigh fading defined by:

$$p_Z(z) = \frac{z}{\sigma^2} exp(-\frac{z^2}{2\sigma^2}) \tag{5.19}$$

A more general fading distribution was developed whose parameters can be adjusted to fit empirical measurements. This distribution is called the Nakagami and is given by:

$$p_Z(z) = \frac{2m^m x^{2m-1}}{\Gamma(m) P_r^m} exp(\frac{-mz^2}{P_r}) \tag{5.20}$$

The Nakagami distribution is parametrized by P_r and the fading pa-

rameter m. For $m = 1$ it becomes Rayleigh fading, instead for $m = \frac{(K+1)^2}{2K+1}$ the distribution is approximately Rician with parameter K.

5.2 Experimental set up

The scenario considered to validate the proposed methodology consists of two intelligent vehicles called iCab (Intelligent Campus Automobile), shown in Fig. 5.3b, with the capabilities of autonomous driving [66]. The following experiments were conducted in collaboration with the Intelligent Systems Laboratory, Department of Systems Engineering and Automation of the University Carlos III de Madrid, Spain. The vehicles are equipped with different sensors such as one lidar, a stereo camera, and encoders. The iCab vehicles follow the same movement trace marked as *red* in Fig. 5.3a, keeping their position one after the other. For this reason, the iCab1 vehicle is called *leader* and the iCab2 vehicle is called *follower*. The dimension of the movement trace in the testing environment is $38mX33m$. Information about the control of the vehicles, i.e., steering angle (s) and power (p), along with the odometry data (x and y positions), are the data exchanged during the operative process. The sampling rate of the sensory observations is 0.33 seconds. After collecting the datasets, a synchronization operation is performed in order to perfectly synchronize the collected datasets.

(a) Testing environment. (b) iCab platforms.

Figure 5.3: The vehicles and the environment used for the experiments.

To test the anomaly detection model, we used the vehicles' data while performing two different actions in the same test environment. The

scenarios are:

- Scenario I *Perimeter monitoring*: iCab vehicles perform platooning operation in a closed environment, as shown in Fig. 5.6. In total, four laps (i.e., the platooning operation has been performed four times, one after the other) have been performed and collected the data. The follower vehicle mimics the actions of the leader vehicle. This is the scenario used in the training phase to learn the switching DBN models.
- Scenario II *Emergency stop*: while both vehicles are moving along the rectangular trajectory of the perimeter monitoring task, a pedestrian suddenly crosses in front of the *leader* vehicle. As soon as the *leader* detects the presence of the pedestrian, the vehicle automatically executes an emergency brake and waits until the pedestrian crosses, and then continues the task. At the same time, the follower detects the anomaly in the state of the leader, and it also performs a stop operation until the leader starts its movement again. The datasets from this scenario have been used to test the switching DBN models learned in the training phase. We have used about 30% of the datasets to test the learned DBN models.

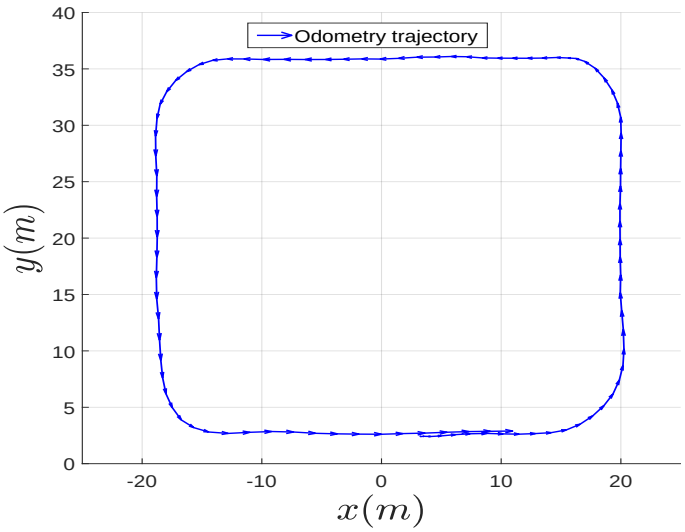
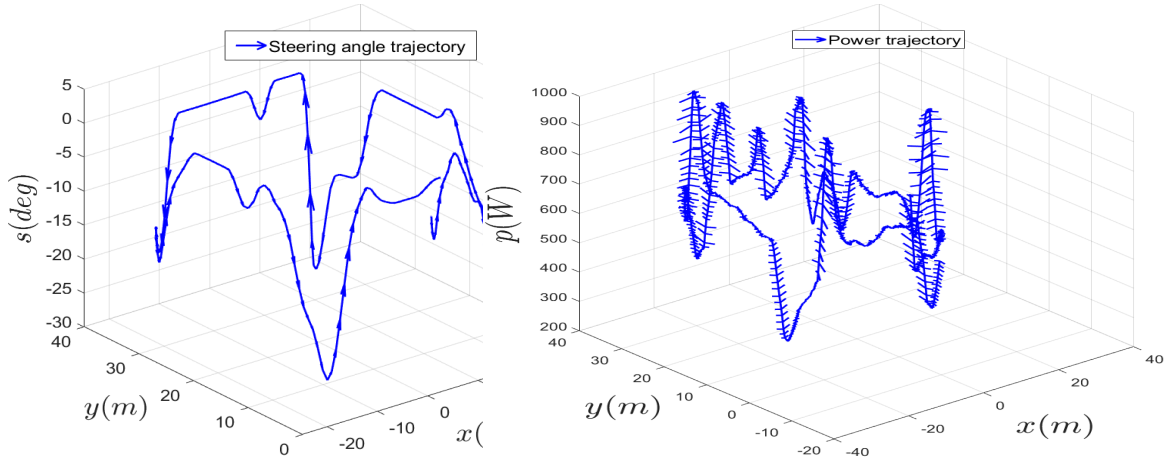


Figure 5.4: Position data for perimeter monitoring task.



(a) Steering angle w.r.t position. (b) Power w.r.t position.

Figure 5.5: Control variables plotted w.r.t position.

Fig. 5.4 plots the one lap (from the four laps) odometry (x and y positions) data for the perimeter monitoring task. Fig. 5.5 shows the steering angle w.r.t the vehicle's position (Fig. 5.5a) and the rotor power w.r.t the vehicle's position (Fig. 5.5b). For simplicity, in Fig 5.5, plotted only the data from one lap (i.e., about 800 data points).

To test the reliability and quantify the expected delay of the data exchange between the two vehicles, we have used the ONE simulator [55]. It is a network simulator designed for testing communications among moving objects. The real trajectory data (of size 38mX33m) of the PM task (refer Scenario I in Fig. 5.6) has been doubled the size i.e., 76mX66m and is inserted in the simulator as the Well Known Text (WKT) file format and created two dynamic nodes that represent the header (iCab1) and follower (iCab2) vehicles. The trajectory has been doubled inorder to make the simulation closer to real situations.

Moreover, we have analyzed how the packet loss and delay affect the proposed learned DBN models for abnormality detection. The delay for data communication in an ego-things network is defined as the time taken to transmit a data packet from the sender ego-thing to the receiver ego-thing through the transmission medium.

Considering the current state-of-the-art, the IEEE 802.11p protocol is one of the most feasible and widely considered in the inter-vehicles

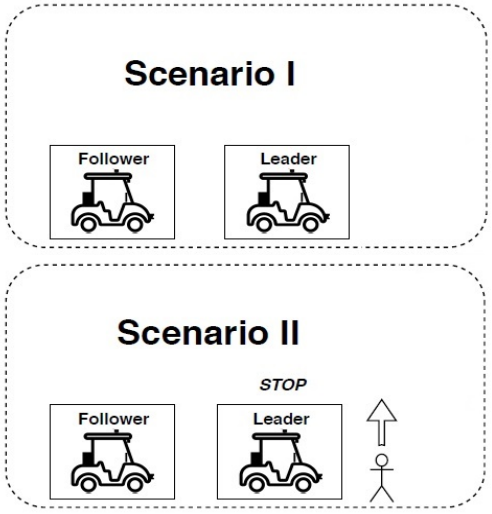


Figure 5.6: Scenarios considered for the anomaly detection test.

communication scenario, especially in the autonomous vehicle networks [97].

A new interface has been created in the ONE simulator in order to be able to model the inter-vehicle channel as a Rician channel and to set different values for its parameters, including transmitted power, central frequency, and Rician K factor.

We assumed that the data to be communicated between the ego-things are: XY position, steering angle (s), and power (p) of the rotor of the iCab vehicles together with a time stamp. In this way, we assume that the amount of data to be sent is 4 Bytes for the position + 2 Bytes for the steering angle + 2 Bytes for the rotor power + 4 Bytes for the time stamp. The total data payload size is 12 Bytes. Assuming UDP, IP, and IEEE 802.11p as transport, network, and data link layer protocols, respectively, the overall size of each data packet is 12 + 8 + 20 + 28 + 6 = 74 Bytes. The total number of data packets that need to be transferred to the wireless channel during the online test phase is 800 (with each data packet size of 74 Bytes.)

5.3 Results

Results of offline learning of DBN models are here not described in all their steps but just providing some general overview. Then, the application of models learned in the online test phase is described in more detail to highlight their application to the agents of the ego-thing network.

5.3.1 Offline model learning phase

In order to collect training and test datasets, the vehicles performed platooning operations four times, one after another. The size of each data sequence is about 3200 (i.e., 800 samples per each). The null force filter [27] with a single switching variable corresponds to a unique dynamic model that assumes no state change produces an error sequence when the data sequence violates this rule. The DBN model in the discrete level (i.e., word level) has been learned separately for each ego-thing while they were doing the same cooperative task. The initial filter [27] has been applied to all the agents in the network. The error sequence produced by this initial filter has been clustered to define state-dependent linear dynamic models characterizing the state as varying according to average derivatives and their covariance.

The total number of clusters (nodes) obtained by clustering the states and errors (obtained from the initial filter) was 35 for state space and also 35 clusters for state derivatives with corresponding dynamic models. The GNG reached convergence in this number. Each node cluster corresponds to a letter with respect to respective vocabularies, and the word list has been composed of the different possible combinations of letters from state and state derivative vocabularies (i.e., switching variables and related dynamic models). Then a unique label is given to each letter combination, and at last, 442 such unique combinations of letters (i.e., words) were kept. A transition matrix at the discrete level was then estimated for each agent whose size was $442X442$. This information constitutes the DBN model of the MJPF filter to be applied to each agent.

5.3.2 Online test phase

In the online test phase, the dataset of scenario II, i.e. emergency stop (refer Sec. 5.2), has been employed to check the prediction capability of switching DBN models learned in the training phase and to detect the presence of abnormal situations in the environment. The model was able to detect the abnormality situation due to the emergency brake obtaining high values of the Hellinger distance metric, as shown by *cyan* shaded area in Fig. 5.7 (iCab1 - leader) and 5.8 (iCab2 - follower).

As can be seen in both figures, there is a significant rise in the Hellinger distance abnormality measures during the abnormality intervals. However, the abnormality peak of the follower vehicle is not as high as the leader's. The main reason is that after the emergency stop of the leader, the follower gradually decreased its speed rather than doing an emergency brake. We set the abnormality threshold to 0.4 (indicated by the blue dotted line in Fig. 5.7 and 5.8) considering the average Hellinger distance value of 0.2 when vehicles operate in normal conditions.

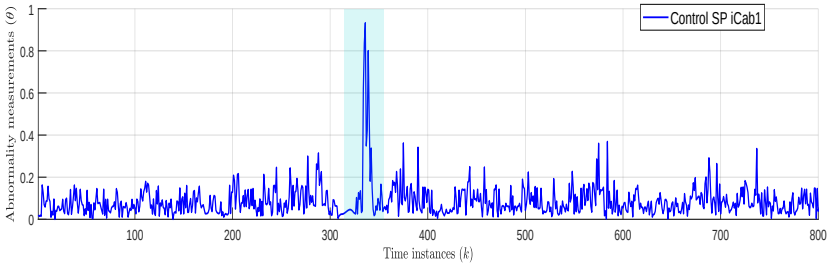


Figure 5.7: Abnormality measurements plot for iCab 1.

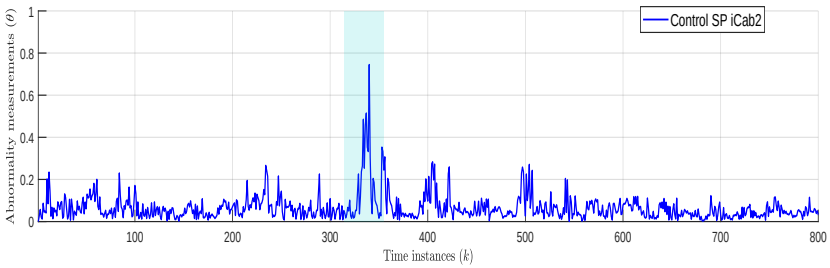


Figure 5.8: Abnormality measurements plot for iCab 2.

As described before, both vehicles have their own DBN model as well as the model for other vehicles. We have shown the DBN model's performance for the leader vehicle in the leader itself and the follower by plotting ROC curves and comparing AUC and ACC parameters. The model for the leader in the follower vehicle can predict the future states of the leader vehicle and detect any abnormal situations present in the environment around the leader vehicle. Once the follower vehicle detects the abnormal situation of the leader, it should adapt its own behavior by changing its future action by appropriate decisions. It should be pointed out that in this work, the focus is on abnormality detection as a basic step of collective awareness, while the impact on such additional information on decision making and online learning of new actions will be made in the future. The important factor that needs to be considered here is the effect of the communication channel over the transmitted data between the vehicles. Such transmission loss causes the degradation of performance of the DBN model and consequently the abnormality detection capability as well.

The DBN model for the leader vehicle inside the follower vehicle estimates the abnormality situation of the leader after receiving real-time observed data (i.e., steering angle and power) from the leader over the wireless channel. Due to the impact of the communication channel over the transmitted data, the DBN model performance decreases, and we have investigated how it affects the capability of detecting abnormal situations. The performance measure we used in this work is ROC curve parameters such as AUC and ACC. The main factors that affect the transmission loss are the data rate of different modulation schemes, the distance between the vehicles, and the Rician K-factor. The IEEE 802.11p standard operates at 5.9 GHz central frequency,

Table 5.1: Simulation parameters.

Data rate (Mbits/sec)	Modulation	Sensitivity (dBm)	K factor
3	BPSK	-85	0,1.8,2.6,3
9	QPSK	-80	0,1.8,2.6,3
18	16QAM	-73	0,1.8,2.6,3
27	64QAM	-68	0,1.8,2.6,3

offers 10 MHz bandwidth, and allows sending data with different modulations, and data rates range from 3 to 27 Mb/s [1]. We have fixed a maximum communication range to 100 m, considering that the data loss is considerable and beyond a possible realistic reliability requirement if the distance is higher. Table 5.1 summarizes the simulation parameters [18] we have used. Different tests have been conducted by changing the values of data rate, modulation, and K-factor to evaluate the model's performance. High K-factor values refer to rural scenarios where the presence of obstacles, buildings, etc., has a lower impact on the achieved performance. The sensitivity column shows the minimum values of the signal-to-noise ratio (SNR) at the receiver to guarantee successful data reception [18].

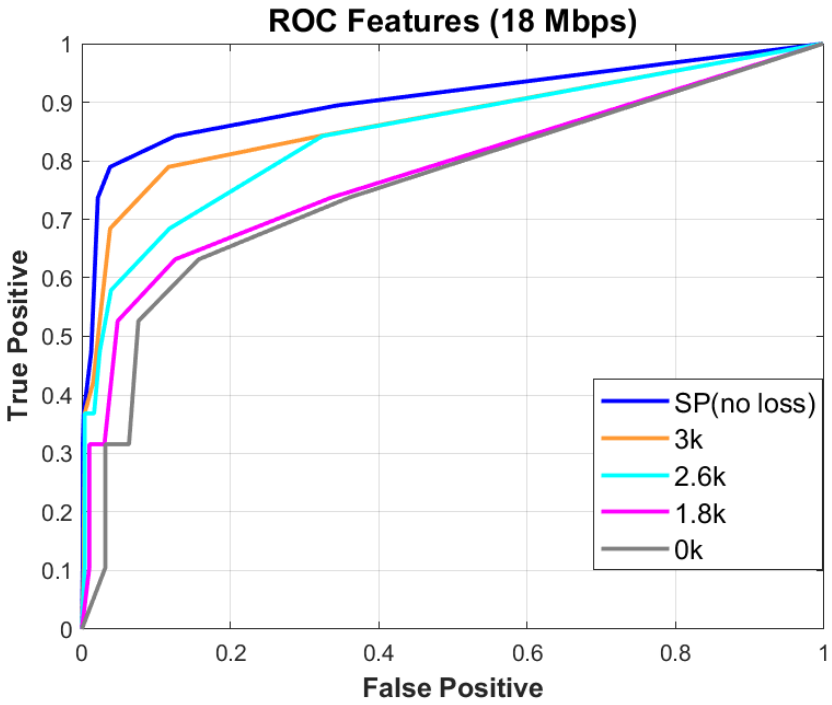


Figure 5.9: Receiver operating curve (18 Mb/s)

The DBN model performance in terms of ROC curve has been plotted for the leader vehicle for data rates 18 Mb/s in Fig. 5.9 and 27 Mb/s in Fig. 5.10, respectively. These figures show the reliability of the communications in different scenarios, from completely rural ($K = 3$) to urban ($K = 0$). The blue curve refers to the case without transmis-

sion among ego-things, i.e., the ideal case of complete knowledge, and has been inserted as a comparison.

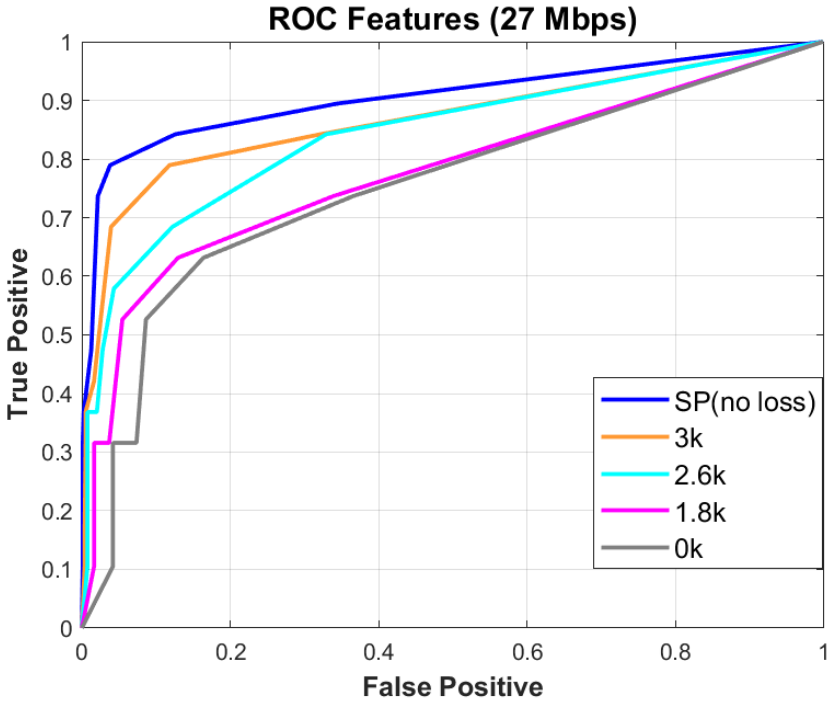


Figure 5.10: Receiver operating curve (27 Mb/s)

When the data rate is 18Mb/s, our learned DBN model performance is not degrading much compared to the no transmission loss case. The performance is in the acceptable range, and the model well predicts abnormal situations. The performance degradation (in terms of AUC) and the accuracy in prediction (in terms of ACC) for 18 Mb/s and 27 Mb/s data rates are summarized in Tables 5.2 and 5.3, respectively. When the environment changes from rural to suburban to urban, the performance of the model again degrades. Finally, when in case of no line of sight component (LOS) ($K = 3$), the value of AUC and ACC in the ROC curve is further reduced.

Table 5.2: ROC features: Data rate = 18Mb/s

K-factor	Area under the curve(AUC)	Accuracy(ACC)
No loss	**0.9039**	**0.9826**
3	0.86665	0.9814
2.6	0.8444	0.9814
1.8	0.7788	0.9764
0	0.7059	0.9764

Table 5.3: ROC features: Data rate = 27Mb/s

K-factor	Area under the curve(AUC)	Accuracy(ACC)
No loss	**0.9039**	**0.9826**
3	0.8653	0.9801
2.6	0.8409	0.9777
1.8	0.7743	0.9764
0	0.6994	0.9764

Moreover, the distance between the vehicles plays a role in packet losses. To analyze the relationship between distance, delay, and data packet loss, we focused on Scenario I by changing the follower's velocity trace to let the distance among them change during the simulation.

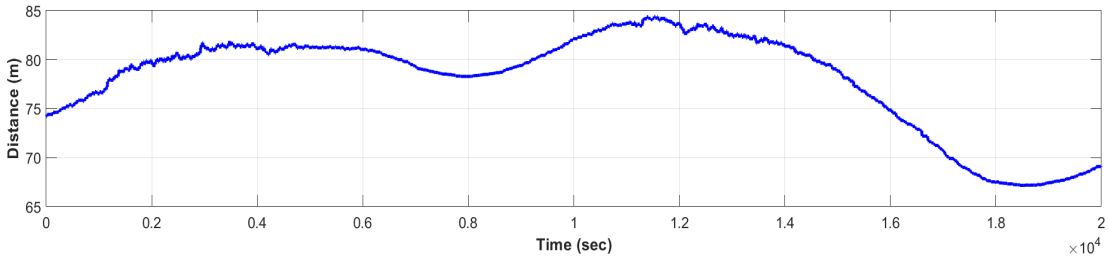


Figure 5.11: Distance versus time.

Fig. 5.11 and 5.12 show how the distance between the two vehicles and the communication delay between them change over time, respectively, while the received SNR (blue plot) and the data packet losses (green dots) are shown in Fig. 5.13. It is evident from the figures that when the distance increases, the delay as well as the packet loss also increases. The maximum delay occurred was 24 μs when the distance

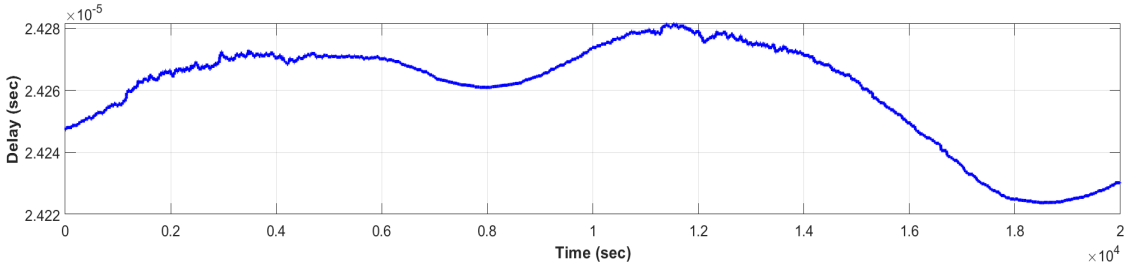


Figure 5.12: Delay versus time.

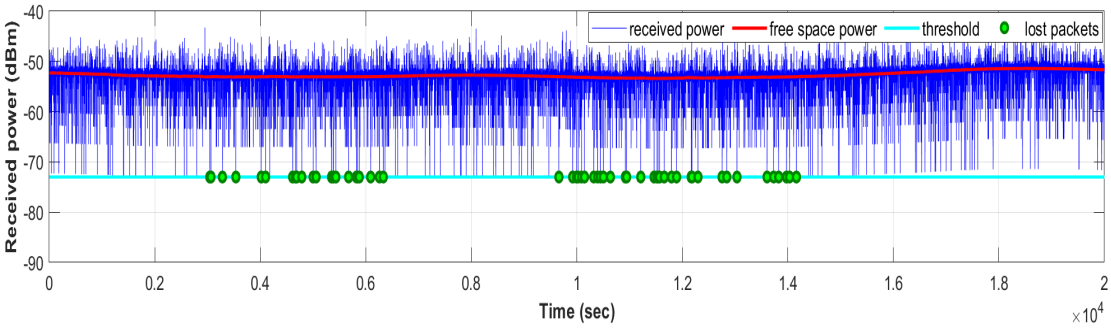


Figure 5.13: Received power and packet losses.

between the ego-things were maximum which is about $83m$. The delay of 24 μs is relevant for the presented approach even though there was a degradation in performance of the model.

Fig. 5.13 also shows the power in free space (red line), and sensitivity threshold (light blue line) of the power corresponds to the data rates we considered and finally, the lost packets as it did not satisfy the threshold limit of the minimum received power. The overall amount of packets lost is shown in Table 5.4.

Table 5.4: Packet loss ratio for different K values and different data rates

K-factor	Packet loss ratio (18 Mb/s)	Packet loss ratio (27 Mb/s)
3	0.0025	0.0037
2.6	0.0037	0.0074
1.8	0.0099	0.0160
0	0.0310	0.0410

Considering the shown results, different considerations can be pointed

out about which data ego-things should exchange to increase the reliability of the described system. For example, if the current distance between vehicles allows obtaining a packet loss ratio below a certain threshold, the vehicles can decide to communicate the ground truth observations to the other vehicles in the network to better detect if there are any abnormalities in the environment. Otherwise, if the vehicle approaches to the border of transmission range or the distance between them exceed a certain threshold, it would be more appropriate to communicate only the abnormality measurements as soon as it detected rather than communicating all the ground truth observations. The transmission loss is directly proportional to the distance, such that if we transmit more data the loss also increases. To reduce the impact of false alarm or missed detection in sensing the abnormal situation, in such situations (higher distances), communicate only abnormality measurements could be more appropriate to give an indication to other ego-things in the network. Although in small distances it is recommended to exchange the observed data itself to detect abnormalities with an acceptable delay.

5.4 Chapter summary

This chapter proposed a method to recognize abnormal situations in smart object networks. Each entity learns a set of DBN models describing the normal behaviour of itself and all the other entities in the network. The considered abnormality metric is based on the Hellinger distance between objects. A MJPF is employed to infer the future states of the entities.

The abnormality metric values calculated in each of the DBN models suggest that our method provides good performance in detecting the environmental abnormalities. Moreover, information exchange among entities has been considered in order to enhance the proposed strategy.

The considered test scenario is composed of two smart vehicles, one (the follower) following the other (the leader), which move along a predefined track. Communication performance has been collected in order to verify the reliability of the data exchange, quantify the ex-

pected performance in terms of delay and loss and consider how these performances could affect the abnormality detection process. We investigated the DBN model performance in the case where each object communicates the ground truth observations to the other entities in the network. To compare the performance with different parameters of the considered channel model (Rician model), such as K-factor, distance, and data rates, we have plotted ROC curves and calculated reliability (AUC) and ACC metrics.

Chapter 6

Implementing interactive Collective Awareness in a network of Ego-things: Phase II

The previous chapter (Ch: 5) presented a method to develop an initial level of collective awareness (CA) in a network of intelligent agents. Moreover, to assess the developed models' reliability and accuracy, different channel conditions such as packet loss, delay, data rate, environment, etc., are considered. IEEE 802.11p protocol is used for the communication among the ego-things to exchange data. To develop a collective awareness model, only considered a single modality of the ego-things. The models are learned separately for each ego-things in the network, and the metric used to estimate abnormality is Hellinger distance. The ROC curve plotted and measured AUC and ACC to compare the models' performance and check whether it comes under the acceptable range or not.

This chapter (Ch: 6) is an extension of the work proposed in the previous chapter, and it is focused on developing multimodal collective awareness models by treating the sensory data from all the ego-things in the network. In addition, this chapter considered two communication protocols such as IEEE 802.11p and IEEE 802.15.4, to compare

the models' performance. The model performance evaluation metrics used are MSE, ACC, and F1-score.

The main objective of this chapter is to develop multi-modal CA of multiple networked intelligent *agents*. Each agent is here considered as an Internet of Things (IoT) node equipped with machine learning capabilities; collective awareness aims to provide the network with updated causal knowledge of the state of execution of actions of each node performing a joint task, with particular attention to anomalies that can arise. Data-driven dynamic Bayesian models learned from multi-sensory data recorded during the normal realization of a joint task (agent network experience) are used for distributed state estimation of *agents* and detection of abnormalities. A set of switching DBN models collectively learned in a training phase, each related to a particular sensorial modality, is used to allow each *agent* in the network to perform synchronous estimation of possible abnormalities occurring when a new task of the same type is jointly performed. Collective DBN (CDBN) models are associated with a Bayesian inference method, namely Distributed markov jump particle filter (D-MJPF), employed for joint state estimation and abnormality detection.

The effects of networking protocols and of communications in the estimation of state and abnormalities are analyzed. Performance is evaluated using a small network of two autonomous vehicles performing joint navigation tasks in a controlled environment. In the proposed method, firstly, the sharing of observations is considered in ideal condition, and then the effects of a wireless communication channel have been analyzed for the collective abnormality estimation of the agents.

The main contributions can be summarized as follows:

– A method is proposed to learn models from low dimensional data sequences representing collective awareness of a network of intelligent entities. For the inferences, a MJPF based on generalized DBN models is used and extended to become able to detect abnormalities at different abstraction levels. An analysis performed to show how the learned models can not only provide a global estimation of anomalies based on the whole set of multidimensional variables (used in the models) but can also provide an explain-

able insight of anomaly related to single specific components of the model. This is considered as an additional explainability feature of the model.

– The robustness of the distributed abnormality detection feature of models concerning a realistic communication channel model is investigated. Evaluated the performance in order to estimate, on the one hand, the reliability and accuracy of abnormality detection under the hypothesis of perfect communication (i.e., no data loss and transmission delays), and, on the other hand, analyzed the robustness of the system model against packet losses and transmission delays of the communication channel among objects by considering different protocols and channel conditions.

6.1 Design and Implementation

The data-driven method introduced to learn collective awareness models for a network of ego-things considers low dimensional multimodal sensor data. The selection of modalities is related to the tasks to be performed; generalized features observed by ego-things should be capable of representing the dynamics of the interaction they have with the environment and other objects during the task. Low-dimensional data here consists of exteroceptive sensor data related to the position of entities in an environment and two different combinations of proprioceptive control information that are causally connected to the motion (i,e, derivatives of position) of ego-things.

Dynamic Bayesian data-driven model learning is used for abstracting at different levels of dynamic rules driving the collective behavior of a group of ego-things in a training phase. Ego-things' training happens when they perform a connected and collaborative specific task. The model learning process initially performed consists of the estimation of a generative model and its component pieces. This includes learning continuous dynamic conditional probabilities, semantic vocabularies at discrete levels. The necessity of establishing in the model condition bidirectional probabilities among exteroceptive and proprioceptive DBNs also implies estimating co-occurrence matrices.

The generative nature of DBN models allows probabilistic inference methods to be defined that differ also depending on the type of DBN chosen to represent data. For example, KFs, HMMs, and PFs can be used for Bayesian inference on simpler DBNs containing only observation and hidden state nodes [13, 88]. A MJPF [31, 33] works instead of on DBNs with three levels of variables, i.e., including discrete switching variables. MJPF Bayesian inference can enrich to make available beyond prediction and state joint estimates at discrete and continuous levels, also probabilistic anomaly estimation as a Self-awareness component.

In collective awareness, multiple DBNs related to multiple agents should become coupled to represent collective interactions. In this case, inference methods like MJPF can still apply, despite the inference steps have to manage the higher complexity of prediction models. In this latter case, as agents on the field sparsely collect observations used by MJPF, the important aspect is determining the impact of using communication schemes to share such observations among agents in the network. Sharing allows collective awareness to be possible simultaneously in each agent in a distributed way. Effects of the wireless channel due to packet loss over the model performances so become important to be modeled and analyzed as done in this chapter.

The description of the proposed method is divided into two parts: learning of collective awareness models (offline phase) and testing the fitness models (online inference phase) and the following sections explain the various steps involved in the process.

6.1.1 Collective awareness model learning (Offline)

This work considers three different modalities, each of them able to capture a part of the essential information necessary to provide CA to the network of ego-things. The possibility to estimate direct causal relationships between the environment state and the network of agent states when they perform a co-operative task is considered to select such modalities. The model learning steps are the same for all the modalities and assumed that the multimodal acquired data sequences are available for all the networked ego-things. The block diagram

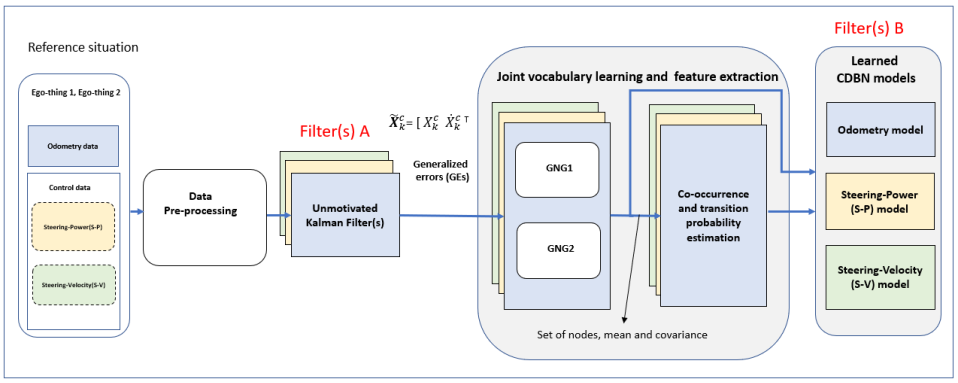


Figure 6.1: Block diagram: training phase.

representation of the training phase is shown in Fig. 6.1.

Pre-processing and estimation of generalized errors (GEs)

Once the sequences of multimodal data samples are available, time alignment is performed to match their timestamps. The first level of synchronization occurs between heterogeneous data of every single ego-thing (intra-synchronization). Ad-hoc inter-synchronization is also necessary among the data collected by different ego-things that are part of the considered network as they can be of different clocks. Three sensor modalities here considered are odometry (X-Y positions) as exteroceptive data, control steering angle-power *(S-P)* and control steering angle-velocity *(S-V)* as proprioceptive data. The chosen sensor data are low dimensional, i.e., each provided a 2D vector of observations for a single ego-thing. In this chapter, with no lack of generality, a network of two ego-things is considered to provide experimental results on collective awareness, so 4D sensor data sequences are used for collective model learning for each modality.

Let $Z_k^{(e1,e2,\ldots,en),c}$ be the measurements from all the ego-things related to modality 'c' at the time instant k and $X_k^{(e1,e2,\ldots,en),c}$ be the associated joint latent state variables. The measured observations can be mapped to the latent states by the following observation model:

$$Z_k^{(e1,e2,\ldots,en),c} = g(X_k^{(e1,e2,\ldots,en),c}) + \boldsymbol{\epsilon}_k \tag{6.1}$$

where $\boldsymbol{\epsilon}_k$ represents the vector composed of measurement errors (for each ego-things) at a time step k. g() is a function that, in this chapter, is assumed to be linear.

This assumption meets easily by considering low-dimensional exteroceptive and proprioceptive sensory data to learn switching DBN models. The sensors can design to calibrate to acquire these features around working points, so allow statistical linearization of the relation between observation and hidden, latent variables. This assumption here allows the work to focus on non-linearities in the dynamic models that DBNs can learn through switching models and can be related to the agent's capability to predict and detect anomalies in their dynamic behaviors. However, the proposed method already proved to be extendable to situations where it used high dimensional data (like videos) and non-linear observation models. In this case, the problem is more complex because sometimes the function g() is also not known as has to be estimated jointly with prediction components of the generative model. Tools like Generalized Adversarial Networks (GANs) and Variational Autoencoders (VAEs) have to be integrated into the DBN to map observed sensory data into generalized state variables of the DBN model [93, 82]. This goes beyond the scope of this work, as explained.

Learning a DBN model is a recursive incremental process. In fact, the input information to the learning steps consists of a state grounded set of deviations of observed data from predictions provided by the inference process associated with an already existing generative model, i.e., an initial DBN. Generalized error is used as the definition of an error that consists of coupled information, including a state and the deviation found in the state for higher-order derivatives (e.g., in this chapter limited to first-order state derivative). The detection of generalized errors (e.g., a mismatch between Bayesian predictions and updates) is done through anomaly detection. At the same time, learning is a process of finding a new DBN that minimizes the presence of GEs in a given sequence. It showed that an initial reference generalized filter [27] could be applied low dimensional data to produce generalized errors. In this chapter, this is done with each modality data sequence to produce the generalized errors (GEs) from which to learn the task

model.

The initial model is based on a null force filter (Unmotivated Kalman filter in this work) that assumes the absence of forces between consecutive time instances. It is equivalent to suppose that the agent's state vector at a time instant $k+1$ will remain unchanged w.r.t the previous time instant k except for low variance Gaussian perturbations. When the observed data series do not follow this rule, record the derivatives' errors in a given position (i.e., generalized errors associated with anomalies) and post-process to learn a new model. This corresponds to obtain a new DBN where the associated inference model (in our case a MJPF) will generate minimal GEs if applied to similar sequences as those from which the model was learned.

The generalized error (GEs) related to modality 'c' can be written as:

$$\tilde{\mathbf{X}}_k^c = [X_k^c \;\; \dot{X}_k^c \;\; \ddot{X}_k^c \cdots X_k^{d,c}]^\intercal, \tag{6.2}$$

where d indexes the d-th time derivative of the state. In Eq. 6.2, all components are random vectors. To describe errors jointly from different entities, one can organize such vectors in different ways. Here, for example, a vector in Eq. 6.2 is described below in Eq. 6.3 that includes all vectors of all the entities for a certain fixed derivative order, i.e., $d = 0$. Other similar vectors can write for higher-level derivatives.

$$X_k^c = [X_k^{e1}, X_k^{e2}, ..., X_k^{en}]^\intercal \tag{6.3}$$

In this work, we have limited the generalized errors to first-order derivatives, and models have been learned accordingly.

Joint vocabulary learning and feature extraction

Once the generalized errors are estimated, the very next step is to perform unsupervised clustering as part of the learning process of the CDBN generative model (refer Fig. 6.4). The probabilistic links that connect variables in the CDBN are also estimated within this process. A hierarchical switching 2-Time Slice DBN (2T-DBN) [58] is chosen as

the Generative model, and it is shown in Fig. 6.4. The model is composed of two levels beyond the observation level: a continuous and a discrete generalized state level. Unsupervised clustering allows learning a semantic vocabulary consisting of clusters of GEs with similar state and derivative values. A different switching variable assigns to each cluster, and this variable represents the discrete switching variable. As each cluster is characterized by its own average derivative, a different linear dynamic model at a continuous level associate with each cluster label, so specifying a further element for the generative model. Such local conditional elements of the generalized model are useful for allowing the model to represent a piece-wise linear dynamic behavior (one way of approximating nonlinear models) for each modality.

The sequential probabilistic trajectories of multiple switching variables, i.e., modes of behavior of the dynamic system, can be represented by transition matrices at the discrete level. Switching models are associated with inference methods: For example, Markov Jump Particle Filters can be seen as composite joint filters, where KFM is used at the continuous level to allow inferences on local linear components of a dynamic model. The PF acts as a second filter on the discrete switching variables to regulate switches among successive elements of the piece-wise linear discrete dynamics).

The learning of a DBN switching model from GEs implies the capability to cluster GEs into groups that show similar properties (similar dynamic linear behavior in state regions). To this end, unsupervised clustering is necessary. The unsupervised clustering approach used to obtain clusters from GEs is the GNG algorithm [39]. The input multimodal GEs data sequences provided to each GNG here consists of GEs computed separately, applying an initial filter to different agents' data collected when performing a collective task. In this chapter, the algorithm used implies separate clustering to be applied to different vectors associated with a given derivative (refer Eq. 6.2). A successive hierarchical clustering step is applied to obtain GEs clustering thanks to synchronization information. Two ego-things are here considered for simplicity so that the input of each GNG consists of a four-dimensional (4D) vector. Therefore, for each modality, two GNGs have to be per-

formed, one for the GE's state component in Eq. 6.2, noted as GE0, and the second for the derivative component GE1 as provided by the initial filter. For example, the input vectors to the GNGs belong to the odometry modality are in the form as below.

$$GNG1, \quad X^c_{1,k} = [x_1 \;\; y_1 \;\; x_2 \;\; y_2]^\intercal \tag{6.4}$$

$$GNG2, \quad X^c_{2,k} = [\dot{x}_1 \;\; \dot{y}_1 \;\; \dot{x}_2 \;\; \dot{y}_2]^\intercal \tag{6.5}$$

The output of each GNG consists of a set of clusters, each one characterized by the mean and the covariance matrix of GEs being attributed to that cluster, so providing an uncertainty based boundary of each cluster. A cluster can see as nodes *nodes* of a graph of switching variables. Each node groups a subset of samples of GEs (i.e., GE0 or GE1) that have a low distance wrt the mean of the region associated with the node. The nodes produced by GNGs are the discrete components or switching random variables of the CDBN model. For instance, the group of nodes created by a GNG of modality 'c' of l^{th} order time derivative vector of GEs written as:

$$\boldsymbol{S}^{c,l} = \{S_1, S_2, \ldots, S_m\}, \tag{6.6}$$

where m represents the maximum number of nodes produced by the GNG. An example of the group of nodes produced by the GNG belongs to ego things states are shown in Fig. 6.2. Each row represents the cluster centroid or node produced by the GNG. In this example, the total nodes produced is 35, and the dimension is 4D.

The co-occurrence matrix requires a further post-clustering step taking into account relationships between different GE spaces to find the temporal correlation between the switching variables. The nodes activating at the same time instance from GE0 and GE1 discrete cluster spaces are grouped as part of the hierarchical successive clustering step to form *words*. A word can be represented as:

$$W^c = [S^0_i \;\; S^1_j]^T \tag{6.7}$$

where S_i represents the i^{th} element of the group of nodes produced

	1	2	3	4
1	-0.1777	-0.0116	-0.1841	-8.2521e-04
2	0.1762	0.0019	0.1993	0.0079
3	4.8918e-04	-0.2175	0.0051	-0.2209
4	-0.0054	0.1871	-0.0011	0.2081
5	-0.1165	-0.0919	-0.1499	-0.0372
6	0.1179	0.0367	0.1344	-1.0579e-04
7	-0.0405	-0.1448	-0.0972	-0.1185
8	-0.1237	0.0864	-0.0298	0.1451
9	0.0787	-0.1040	-0.0043	-0.1666
10	0.0494	0.1601	0.1359	0.0677
⋮	⋮	⋮	⋮	⋮
28	-0.0918	-0.1241	-0.1321	-0.0770
29	-3.1422e-04	0.1813	-0.0021	0.1755
30	-0.0553	0.1179	-0.0181	0.1552
31	-0.1803	-0.0056	-0.2139	-0.0019
32	0.1377	-0.0863	0.0729	-0.1091
33	0.1879	0.0069	0.1984	-0.0060
34	0.0038	-0.1973	-0.0047	-0.1992
35	0.0884	-0.1228	0.0458	-0.1472

Figure 6.2: Example of the group of nodes produced by the GNG clustering of ego-things' states.

by GNG1 belongs to GE0. Likewise, S_j^1 represents the j^{th} element of the list of nodes produced by GNG2 (i.e., the GNG belongs to GE1 space).

In Fig.6.3, the first two columns represents the coupled nodes belong to GNG1 and GNG2 respectively. Total 114 unique coupled nodes produced and for each couple assigned a label. A unique label assigned for each word or combination of nodes and the complete list of formed words is called *dictionary*. The resulting dictionary is:

$$\boldsymbol{D}^c = \{(W^{(1),c}\ L_1), (W^{(2),c}\ L_2), \ldots, (W^{(m),c}\ L_m)\}, \tag{6.8}$$

where L represents each word's unique label, m represents the index of the maximum number of elements in the dictionary. An example of dictionary is shown in Fig.6.3. The dictionary information is used in MJPF for the joint prediction and estimation of future states of the

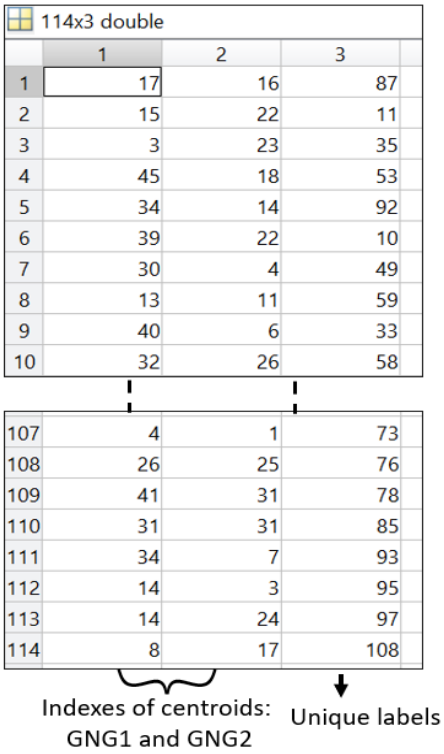


Figure 6.3: Dictionary or group of words generated from the coupled nodes.

ego-things.

Then the co-occurrence probability matrix has been estimated from the complete list of words. It provides the information about the GNG nodes enable in GE1 space corresponds to nodes in GE0 space. The co-occurrence matrix for modality c can be represented as below:

$$T_c = \begin{pmatrix} \theta_{11} & \theta_{12} & \dots & \theta_{1n} \\ \vdots & \ddots & & \\ \theta_{m1} & \theta_{m2} & \dots & \theta_{mn} \end{pmatrix} \tag{6.9}$$

The rows of the matrix 1-m called transitional elements correspond to the total number of nodes generated from generalized error 0 (GE0) by GNG1. Similarly, the columns of the matrix 1-n are absorbing elements represent the total nodes produced by GNG2 of generalized error 1(GE1). Each of the matrix elements θ is an estimation of the probability of occurrence between GE0 and GE1 spaces. For example,

θ_{13} is a co-occurrence probability value between the first node of GNG 1 to the third node of GNG 2. The causal relationships between different GE spaces help extract various features of the discrete cluster level from each modality's viewpoint and summarised below.

- **Odometry *X-Y***: The initial generalized filter produces generalized errors (GEs) from the exteroceptive sensory data of odometry. Then discretized the GEs by GNGs and obtained GE0 and GE1 cluster spaces. The generalized GE0 encodes the location information of the ego-things, and at the same time, GE1 gives focus to the direction of movements. Then, the co-occurrence matrix is estimated; it provides information about the causal relationships between GE0 and GE1 cluster spaces. In other words, for a given node in GE0 space (embed the position information), the co-occurrence matrix tells the possible future direction of movements of ego-things (i.e., the possible nodes enable in the GE1 space) in probabilistic terms. All the information extracted from the GE0 and GE1 clusters, along with the help of the co-occurrence matrix, was used to learn a filter. This filter produces errors or abnormalities when the prediction deviates from the actual measurements. Inside the filter, the spatial features are embedded. Therefore, it can differentiate the types of dynamics of the ego-things based on spatial coordinates in the provided context.

  When an agent network experience different from the one used to learn the filter, it will produce abnormality errors. That means anomaly detection occurs when the existing filter fails to represent the new situation with the knowledge it already has. A new filter will be learned in this situation to embed the knowledge acquired from the current experience. If a similar experience happens in the future, the filter will represent the situation, and the knowledge will help in the joint decision-making of ego-things. This is an evolving concept; more details and results are provided in Section 6.3.1. The extracted feature from the exteroceptive odometry data will enrich each ego-things' contextual awareness in the network. In this level of abstraction, it can detect spatial

anomalies.

- **Control *S-P*** : Contrary to odometry modality, the control *S-P* modality extracts a slightly different feature of the networked system of ego-things. The filter differentiates the agents' types of dynamics, and the location doesn't play any role. The activated GNG nodes in the cluster space of generalized error zero (GE0) during the ego-things' linear movement enable a specific subset of nodes in the GE1 (derivative) space. Similarly, the dynamics in the curved part of the trajectory activates another subset of nodes in both GE0 and GE1 discrete spaces. The filter produces abnormality if the network goes through a different movement pattern than the one used for learning the filter. This feature helps to enrich the self-awareness of each agent in the network.
- **Control *S-V*** : In line with the *S-P* modality, *S-V* modality also identifies the different types of dynamics of the agents' network. Nevertheless, the performance differs based on the joint behavior of the proprioceptive low dimensional data used for learning the filter. When the joint nature of the low-dimensional variables, i.e., steering and velocity, varies while performing a different task and movement patterns, the filter detects the abnormality. This is considered self-awareness property as the used proprioceptive sensory data sequences represent the ego-things' internal behaviors.

Some of the important results of this discrete level filter, along with continual learning, are presented in Section 6.3.1.

Finally, the availability of words (estimated from clusters of GEs) allows the final step to learn the prediction models at the discrete level. To this end, the temporal transition probability between the discrete vocabulary of words can be computed by looking at the relative frequency of time transitions of data. The time sequence is analyzed again to this end to label each observation with words found by clustering and the frequency of changes estimated to complete the DBN model transition probabilities at the discrete switching variable level.

CDBN models

The previous sections present all the necessary steps involved in learning collective DBN models. Each agent in the network learns three CDBN models in total, and each of them represents a particular sensory modality. As states before, all learned models are replica inside each ego-things in the network.

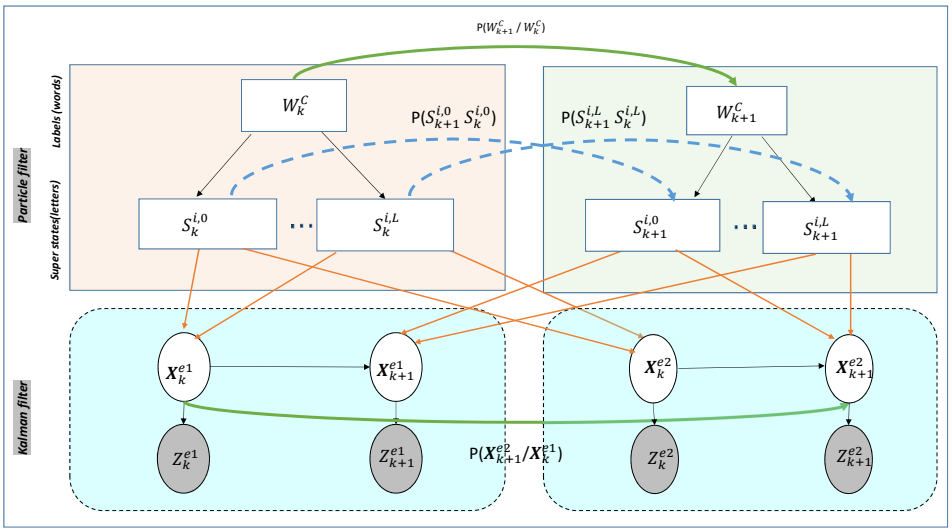


Figure 6.4: A single CDBN model for a two agent network.

The set of CDBN models learned by ego-thing ep and eq is the same for each other ego-thing in the system and can be written as:

$$\boldsymbol{CDBN}^{ep} = \{CDBN^{c1}, CDBN^{c2}, CDBN^{c3}\} = \boldsymbol{CDBN}^{eq}, \forall p, q \in \mathcal{N} \quad (6.10)$$

where c1,c2 and c3 represents the odometry, control S-P and control S-V modalities respectively.

Fig. 6.4 shows the representation of learned CDBN model(by considering two ego-things). The square nodes represent the discrete vari-

ables, *orange* and *green* shaded area represents discrete abstraction level of the CDBN. The round nodes represent continuous variable and are belong to the continuous abstraction level of the CDBN model (i.e., cyan shaded area). The horizontal arrows that are in *green* and *blue* colors represent the conditional probability between two consecutive time instances at continuous as well as discrete levels. Moreover, the vertical arrows (*orange* and *black* in color) describe the causalities between inferences of different ego-things at discrete, continuous states and observation levels.

6.1.2 Model testing (Online phase)

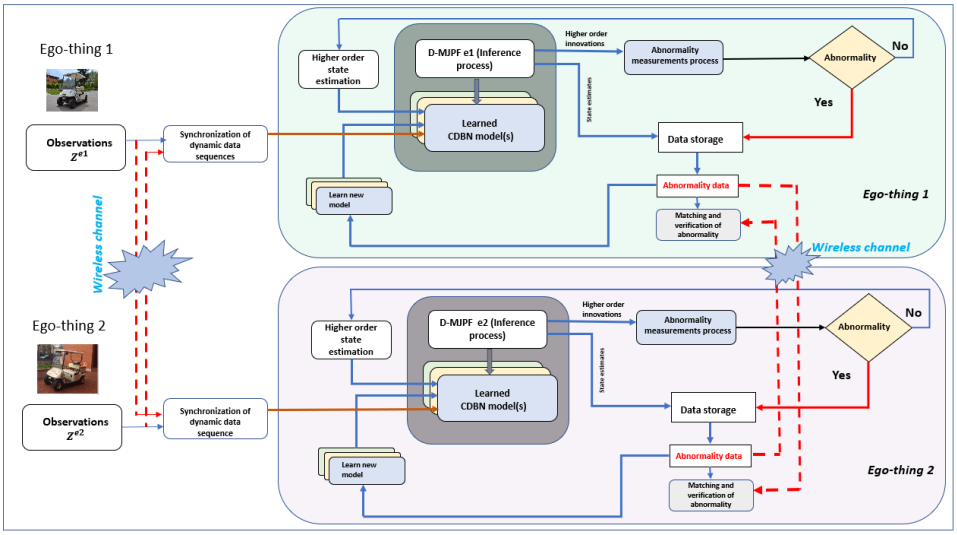


Figure 6.5: General block diagram of CBDN model testing for a two ego-things network. The processes involved in this test phase are common for all the filters learned during the training phase. The red dotted lines indicate communication over the wireless channel.

This part explains the inference process applied to sequences when a given DBN model has been learned and is available. The inference process occurs at different levels of the collective awareness models learned in the training phase. All the filters in this work are using the same method shown in Fig. 6.5, so they can produce new GEs that can be potentially useful for the continual learning of new models. Despite here, we describe a single step in this direction.

The filters produced at the intermediate level (i.e., Filter(s) A in Fig. 6.1) have tested and analyzed the obtained results. The features extracted of the ego-thing by the estimation of the co-occurrence probability matrix used for this purpose. Each of the filters learned in the training phase will pass through the process shown in Fig. 6.5 (during the test phase) to detect anomalies and to learn new filters whenever abnormalities occur. The results of the evolving emergent concept have been presented in Section 6.3.1.

Joint states estimation and abnormality measurements

The process flow diagram of the filter testing is shown in Fig. 6.5. We have proposed to apply a dynamic switching model called Markov Jump Particle Filter (MJPF) [32, 51] to make inferences on the CDBN models learned in the training phase (refer Fig. 6.4). In MJPF, we use KF [100] in continuous state space and PF [43] in a higher hierarchical discrete level. Each dynamic model in the continuous state is associated with one of the discrete set of vocabulary variable. The co-occurrence and transition probability matrices model the switching probability from one mode to another. A detailed description of MJPF is described in Section II of [14]. Here we provide a brief description to understand better how generative DBN models learned can drive the inference process and the related anomaly detection and fixing of GEs.

The objective of MJPF is to iteratively estimate the joint posterior of discrete variables together with continuous states based on an observation sequence. The joint posterior decomposes into a categorical distribution, represented through a set of weighted particles and a set of continuous distributions assumed to be constituted by linear and gaussian variables. The different continuous distribution is associated with different values of discrete variables. A MJPF is an inference mechanism associated with a switching DBN model, consists of prediction/update steps. The particles are predicted using the Gaussian proposal function $q = p(Wk + 1/W_k)$ using Monte Carlo chain concepts associated with a specific algorithm (for example, SIR PF [86]). To this end, the part of the generative model named temporal

transition probability, estimated in the training phase, can be used. However, each particle prediction can perform at the continuous level by the Kalman filter applied to the different linear dynamic models learned during the training phase (using each cluster information).

The propagation of particles allows the prediction of joint continuous/discrete posterior. The update step provides the new observed sensory data sample to the filter; this generates a message passing through the DBN, allowing to update before continuous state variable inside the KF of a particle by means of innovation. Then the message reaches the discrete level where the difference wrt the transition model prediction introduces a new update. Updating allows the global weight of particles to be estimated to represent the new posterior.

In the case of the proposed approach, the posterior probability density function associated with a switching model learned DBN related to modality c is :

$$p({W_k}^c, \tilde{\mathbf{X}}_{\mathbf{k}}^{(e1,e2,...,en),c} / {Z_k}^{(e1,e2,...,en),c}) = p(\tilde{\mathbf{X}}_{\mathbf{k}}^{(e1,e2,...,en),c} / {W_k}^c, {Z_k}^{(e1,e2,...,en),c}) p({W_k}^c / {Z_k}^{(e1,e2,...,en),c}) \quad (6.11)$$

where ${W_k}^c$ is the superstate random variable that represents *words* learned through clustering as the higher hierarchical level vocabulary of switching variables; $\tilde{\mathbf{X}}_{\mathbf{k}}^{(e1,e2,...,en),c}$ represents the joint continuous state of all the ego-things at time instant k.
D-MJPF uses Eq. 6.11 as the target posterior to be iteratively estimated jointly at the discrete and continuous state. The particles' weight is iteratively computed and allows to approximate the posterior. The predicted particles that better match observations obtain the maximum weight, and their positions indicate where the higher probability mass of the posterior is concentrated. D-MJPF can apply to the sensory observed data variables of each modality. However, classical MJPF not having anomaly detection capability. To this end, an additional functionally has to be added to the D-MJPF.

At each time instant, evaluates the distance between prediction and update messages within each level of the DBN of whatever modality, using probabilistic distances like Bhattacharya and Kullback Lieber.

An anomaly is detected whenever this distance is higher than a threshold that reflects the abnormality level definition. Generalized errors correspond to deviations from the prediction found at those continuous states where anomalies present. Therefore a D-MJPF inside each ego-thing can match the currently observed sensory data samples by own sensors and evaluate local abnormalities. The data samples from the other ego-things allow each other ego-thing to perform the same operation on DBNs associated with other agents. The abnormality estimation collectively computed in each ego thing constitutes the collective anomaly detection step.The abnormality information allows each ego-thing to measure how well the learned models fit the currently observed sequence. The anomaly metric used in this work is the filters *innovation* and estimated by the formula:

$$\delta_{k,c} = Z_{k,c}^{(e1,e2,...,en)} - HX_{k,c}^{(e1,e2,...,en)} \tag{6.12}$$

where $\delta_{k,c}$ represents the *innovation* term, $Z_{k,c}^{(e1,e2,...,en)}$ is the observations from all the ego-things that belong to modality c, H is the observation matrix and $X_{k,c}^{(e1,e2,...,en)}$ is the states estimated by the MJPF at time instant k.

In MJPF, we treated the states of all the ego-things together. Each discrete zone has a number of Kalman filters associated with it. The total number of Kalman filters associated with each zone depends upon the number of particles assigned by the discrete level vocabulary. Each Kalman filter will calculate the *innovation* term and average for the estimation of abnormality. Suppose the model detects abnormal situation by testing with dataset from a different experience than the one used in the training phase. In that case, the system stores the abnormality data and then learn a new model from the data. If no abnormality is detected (the *innovation* metric elements are zero), higher-level state estimation will be performed, and the process is repeated. By learning a new model from the abnormality data, the system can represent the new situation. If the system encounters similar experiences in the future, it will infer with the stored representation.

Wireless channel effects over the model performance

The network model of the ego-things is shown in Fig. 6.6. It shows the connections from the sender ego-thing (ego-thing 1) to the receiver ego-thing (ego-thing 2); the same is assumed for the receiver ego-thing to the sender. There are three environmental conditions, such as ideal (no loss), urban and rural, with two protocol standards implemented in the simulator. This work considered only the PHY and MAC layers of the protocols (IEEE 802.15.4 and IEEE 802.11p).

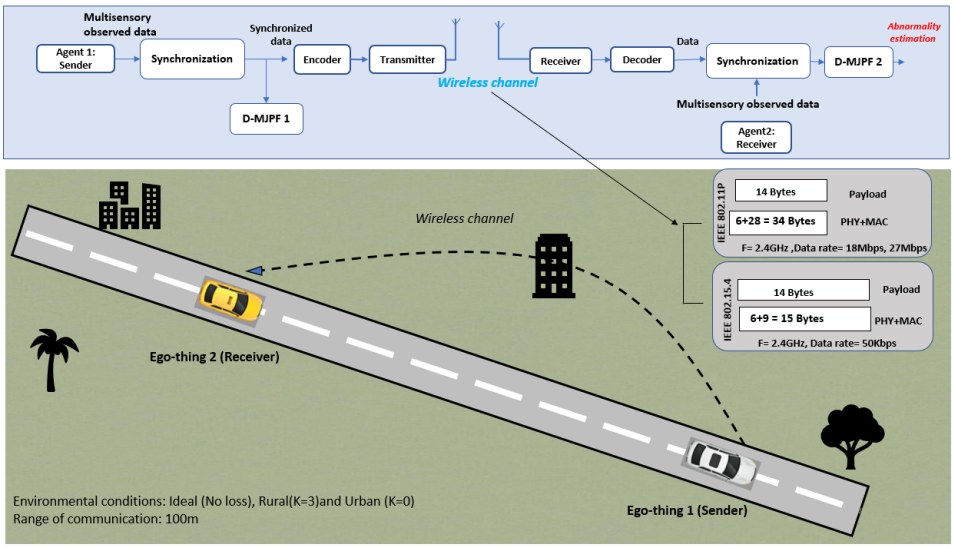


Figure 6.6: A network model for two ego-things. The communication shown is from the ego-thing 1 (sender) to ego-thing 2 (receiver), and the same is for the receiver to the sender.

The throughput of IEEE 802.15.4 is minimal and is less than the PHY bit rate of 50 Kbps. Continuous transmission of packets is not possible as the PHY layer needs to wait for Acks and the CSMA/CA has many timers. By taking into account the PHY layer and MAC layer overheads, the applications have only access to a theoretical maximum of about 50Kbps. Therefore we used a data rate of 50Kbps in this work. When using IEEE 802.15.4, the type of network is an unslotted CSMA for the MAC layer, and the network is PAN with the first node (that starts the network) is the coordinator. Therefore, the connectivity type is ad-hoc if the number of ego-things is more than two. But

IEEE 802.11p supports Device-to-Device (D2D) connectivity among ego-things even if it consists of many IoT nodes.

The data gathered by the exteroceptive and proprioceptive is encoded for transmission over the wireless channel. The encoded data is sent over the channel, the receiver ego-thing collects this data and performs a decoding operation. Then, the joint data (of the sender and receiver ego-things) synchronize to match their data-acquisition time-stamps. A D-MJPF makes inferences, matches the predicted collective states with the observed sensory data to detect the abnormality.

The theoretical analysis of the channel's impact on the data is analyzed in this part. Then the metrics introduced in the next section are used to evaluate the model performance packet loss and delays occurring in the channel.

The wireless channel influences the transmitting data in terms of packet loss, and sometimes a considerable delay in receiving it. Such factors cause a degradation in the performance of the models, and this needs to be analyzed.

In Fig. 6.5, the red dotted lines indicate the data exchange between the ego-things. The D-MJPF will behave differently in situations like lost or delayed packets than how it behaved in an ideal (no loss) situation. The prediction step of the filter will perform jointly, and in the updating step, estimates the posterior for each ego-thing separately. Then estimate the *innovation* metric separately for each ego-thing. For instance, if the ego-thing 1 ($e1$) is not received packets from ego-thing n (en) within an allowed time frame or the packets lost in the channel, the filter will continue prediction based on the previous prior state estimate. Therefore, the co-variance uncertainty increase more until the next observation arrives. As a result, the filter's *innovation* term will become higher during those intervals of packets loss. In case the delay is more than the allowed time, the system will treat this situation as equivalent to lost packets. An example plot of the filter behavior in the presence of lost packets is shown in Fig. 6.7. The confidence interval increases when the model performs the prediction, and no observed data sequences arrive.

The delay and the loss depend on various factors such as the distance

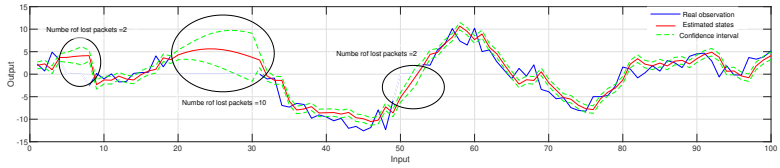


Figure 6.7: Example of model behavior over lost packets. The confidence interval becomes high when the model was not receiving real observations from the agents in the network.

between the ego-things, the communication protocol in consideration, transmission power, the frequency, environmental factors, etc. In this work, we have considered a Rician channel for the study of fading between two ego-things. We have chosen this channel model by considering the distance between the two vehicles not being too high, and the Line of Sight (LOS) component exists between the objects. However, we also investigated the case where no Line of Sight (NLOS) elements exist.

The probability density function of Rician distribution is:

$$f(x|\upsilon,\sigma) = \frac{x}{\sigma^2} exp\left(\frac{-(x^2+\upsilon)^2}{2\sigma^2}\right) I_0\left(\frac{x\upsilon}{\sigma^2}\right) \tag{6.13}$$

where $I_0\left(\frac{x\upsilon}{\sigma^2}\right)$ is the modified Bessel function of the first kind and order zero, υ and σ are the signal strength of the dominant and of the scattered paths, respectively. Rician K factor is:

$$K = \frac{\upsilon^2}{2\sigma_0^2} \tag{6.14}$$

It expresses the ratio between the LOS path power component to the remaining multi-path components. Therefore, υ^2 and $2\sigma^2$ are the average power of the LOS and NLOS multi-path components. As the direct wave weakens, the Rice distribution becomes Rayleigh. The K-factor value zero is equivalent to Rayleigh distribution.

Model performance evaluation metrics

The matching and verification operation was performed after estimating abnormality estimation by the models inside each of the networked ego-things (refer Fig. 6.5).

In all circumstances, models inside each ego-thing can ensure perfect observations from its own sensors, and it can predict its own future state and abnormality measurements without any problems. Simultaneously, the ground truth sensory data sent to other ego-things undergoes the channel effects such as packet loss or delay while transmitting through the wireless channel.

Consider an ego-thing $e1$, the CDBN models inside estimate the abnormality for itself by the ground truth observations collected by own sensors. This abnormality measurement we considered as a reference signal. Simultaneously, the same sensory data from ego-thing $e1$ has been communicated to a second ego-thing $e2$, and the abnormality estimation for ego-thing $e2$ is performed. This time, the transmitted observations were affected by the loss and delay and, consequently, also the models' state prediction and abnormality estimation capability.

To measure the models' performance degradation, firstly, we have estimated the MSE [60] between the reference abnormality signal and the estimated abnormality after the influence of wireless channel. The MSE values presents the discrepancy between the two abnormality signals estimates for the same ego-thing; first one by the models inside itself and the second abnormality signal estimated by the model inside other ego-things in the network. When more packets are lost in the channel or the delay becomes more than expected, the MSE values increases. In the future, the estimation of MSE values can be used to define further the threshold of how much loss the model can accept to assure a certain level of quality in performance.

The formula to estimate Mean Squared Error (MSE) is as follows :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\psi_i - \hat{\psi}_i)^2 \tag{6.15}$$

where ψ_i is the reference signal, and $\hat{\psi}_i$ is the signal to be compared.

In our case, the reference signal (i.e., ψ_i) is the abnormality estimated without delay or loss, and $\hat{\psi}_i$ is the anomaly estimated after the packet loss or delay occurred. The estimated error value tells the reliability of the model for determining an abnormality under the channel's effects. For an in-depth analysis of the model performance by considering the impact of the communication channel, we have considered metrics such as *accuracy* and *F1 score* [80] in addition to MSE estimation. The accuracy is a measure of all the correctly identified samples in the anomaly measurements and is calculated by the Eq. 6.16:

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}, \tag{6.16}$$

Where TP (true positive) is an outcome when the model correctly predicts the anomaly and a TN (true negative) is an outcome where the model correctly predicts the normal situation. Similarly, FP (false positive) is an outcome where the model incorrectly predicts the anomaly, and FN (false negative) is an outcome where the model incorrectly predicts the normal situation.

On the other hand, the F1 score is the harmonic mean of *precision* and *recall* and gives a better measure of the incorrectly classified cases than the accuracy. The estimation formula is:

$$F1 \;\; score = 2 * \frac{(Precision \;\; * \;\; Recall)}{(Precision \;\; + \;\; Recall)} \tag{6.17}$$

where *Precision* is give by

$$Precision = \frac{TP}{TP + FP}, \tag{6.18}$$

and *Recall* can be estimated by the below formula:

$$Recall = \frac{TP}{TP + FN} \tag{6.19}$$

The accuracy metric is used when TP and TN are more important, while the F1 score becomes an important measure when FP and FN are crucial.

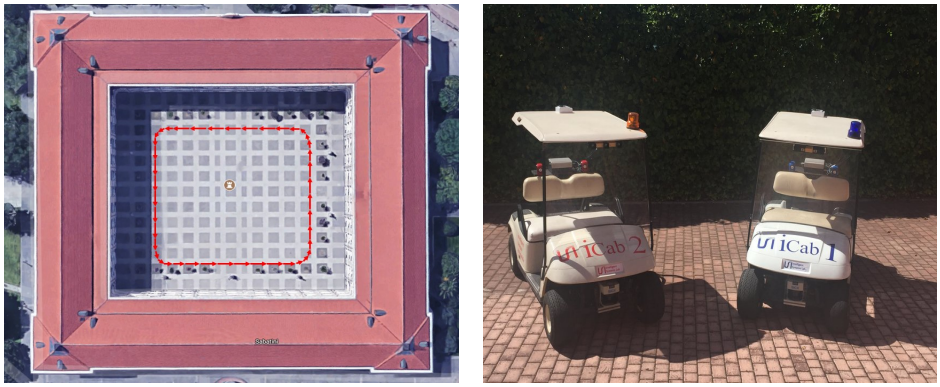
We have used the above evaluation metrics to compare the model performance under the communication channel's influence by considering different protocols and presented the results and analysis in Section 6.3.

6.2 Experimental study

This section explains the case study and the datasets used to validate the proposed methodology. The dataset was collected by conducting real experiments in collaboration with the Intelligent Systems Laboratory, Department of Systems Engineering and Automation of the University Carlos III de Madrid, Spain. Two intelligent autonomous vehicles named iCab (Intelligent Campus Automobile) having the same setup [42] used in this work and shown in Fig 6.8b. Each vehicle is equipped with sensors, such as one lidar, a stereo camera, laser rangefinder, and encoders. This work concentrated on the low-dimensional data of control, i.e., steering angle (s), velocity (v), and power (p), along with the odometry data (x and y positions) of the vehicles. The collected data is synchronized (intra and Ad-hoc inter synchronization) to align their timestamps. The two iCab vehicles perform joint navigation tasks in the rectangular trajectory shown in Fig. 6.8a by keeping their position one after the other with a minimum distance among them. The vehicle navigates in the front called *header* (iCab1) and the one follows is the *assistant* (iCab2).

To train and validate the performance of the collective awareness models, mainly used three low dimensional data combinations such as odometry ($X-Y$), steering-power ($S-P$), and steering-velocity($S-V$) from Scenario I and Scenario II described below.

- Scenario I *Perimeter monitoring (PM)*: The iCab vehicles jointly perform platooning operation in a closed environment, as shown in Fig. 6.8a. The navigation operation performed four times, one after the other, and collected the multi-sensory exteroceptive and proprioceptive data. The *assistant* vehicle (iCab2) mimics the actions of the *header* (iCab1) vehicle.



(a) Testing environment. (b) iCab platforms.

Figure 6.8: The environment and the vehicles used for the experiments.

Fig. 6.9 plots the odometry (x and y positions) data from both vehicles for the Scenario I perimeter monitoring task, blue and red circles indicate the starting positions of iCab vehicles. Moreover, Fig. 6.10 shows the example control signal plots of iCab1 vehicle, and the iCab2 control signals are similar as it mimics the action of the leader vehicle. In Fig. 6.10 (a) and (b), the drop in values happened when vehicle maneuvering in the curves of the rectangular trajectory, and during rectilinear motion, the values of steering and velocity are more steady. In Fig. 6.10 (c) shows the fluctuations in power values during the curved trajectory motion.

– Scenario II *Emergency stop*: While both iCab vehicles jointly navigate in a rectangular trajectory one after the other, a random pedestrian suddenly crosses in front of the *header* vehicle. As soon as the *header* detects the pedestrian's presence, the vehicle automatically executes an emergency brake and waits until the pedestrian crosses and then continues the navigation operation. Subsequently, the *assistant* vehicle (iCab 2) detects the anomaly in the header vehicle and performs an emergency brake operation until the header vehicle starts its movement again. The odometry $(X-Y)$ and control data combinations of steering-power $(S-P)$ and steering-velocity $(S-V)$ from this scenario used to test the fitness of switching CDBN models learned in the training phase.

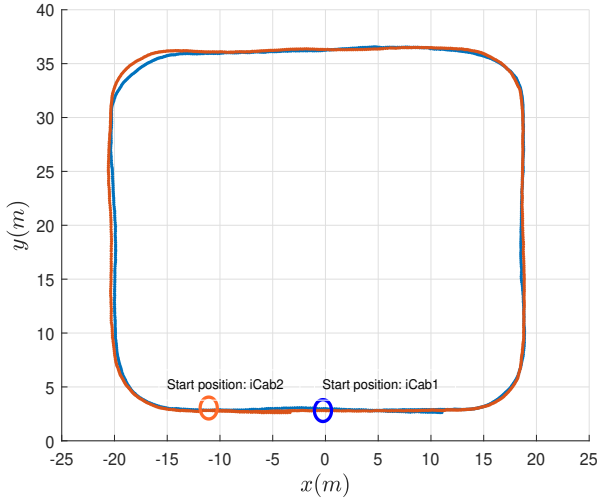


Figure 6.9: Odometry data for perimeter monitoring task (training data).

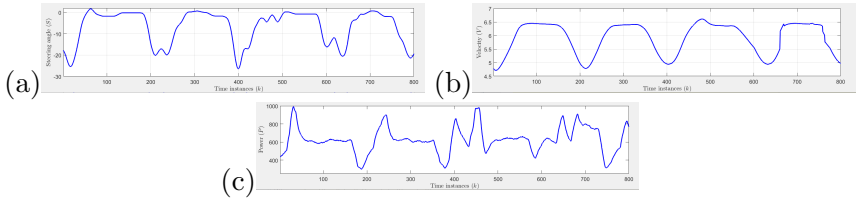
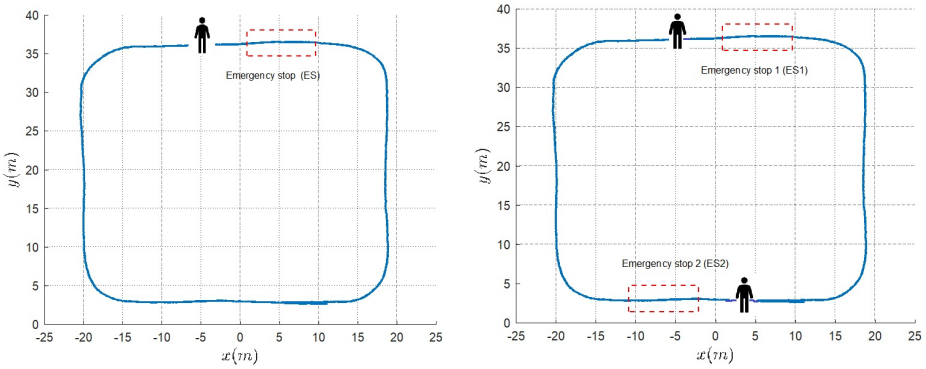


Figure 6.10: Control signal plots of iCab1 (a) *Steering*, (b) *Velocity*, (c) *Power*.

There are two sets of data of three combinations ($X-Y$, $S-P$ and $S-V$) prepared from Scenario II. Fig. 6.11 shows the plot of Odometry ($X-Y$) data. The first one is the emergency brake operation executed once in the complete navigation in the rectangular trajectory called Emergency stop 1(ES1), as shown in Fig. 6.11a). The second dataset is collected while the pedestrian appeared twice, and an emergency stop was performed twice during the platooning operation performed in the rectangular trajectory. This second set of data is named ES2 (Emergency stop 2) and is shown in Fig. 6.11b.

In the real iCab experiments, the vehicles are connected with a base station to exchange data between them, not directly connected. We need an additional simulator for only the connection part to check



(a) Emergency stop 1 (ES1). (b) Emergency stop 2 (ES2).

Figure 6.11: Odometry data of test scenarios.

how the model performance is affected by packet loss and delays happen by the wireless communication channel's influence. For this purpose, we have used a simulated environment to exchange all the sensory data (odometry and two combinations of control data along with their timestamp information) between the ego-things and measured the model's performance by considering various parameters. We have used the ONE simulator in this work [55], and the graphical user interface (GUI) of the simulator is shown in Fig. 6.19. Simulated dynamic ego-things scenarios with two different protocols, such as IEEE 802.11p and IEEE 802.15.4, and compared the performance.

The IEEE 802.11p protocol is one of the most feasible and widely considered standards in the inter-vehicle communication scenario, especially in autonomous vehicle networks [97]. On the other hand, IEEE 802.15.4 is suitable for low-cost, low-speed ubiquitous communication between connected devices [29]. We have used both of the standards in this work and made a comparison of the performances. Additionally, a new interface has been created in the ONE simulator to model the channel between the ego-things as a Rician channel and set different values for its parameters, including transmitted power, central frequency, receiver sensitivity, and Rician K-factor.

The data to be communicated between the ego-things are: $X - Y$ position, steering angle (S), rotor velocity (V), and rotor power (P) of the iCab vehicles with their respective time stamps. In this way, we

assume that the amount of data to be sent is 4 Bytes for the position + 2 Bytes for the steering angle + 2 Bytes for the rotor power + 4 Bytes for the time stamp. By considering only Physical and MAC layers, the total size of each data packet for IEEE 802.11p is 48 (28+6+14) Bytes, and for IEEE 802.15.4 is 29 (9+6+14) Bytes.

6.3 Results

This section presents the results obtained by the proposed methodology applied to the real experimental datasets. Mainly three-level results demonstrated: the first two levels treated the model performance in ideal condition, i.e., without considering channel effects. The final part includes comparing D-MJPF performance with different evaluation metrics by considering two protocols and channel conditions.

6.3.1 Phase 1: Discrete cluster level anomaly detection

The performance of the initial filters (i.e., Filter(s) A in Fig. 6.1) assessed with the ego-things various features learned by co-occurrence probability matrices in this phase. All the filters pass through the processes shown in Fig. 6.5 during the test phase. Scenario II datasets (ES1 and ES2) of different modality used in this part. The detailed analysis of the results is presented only for the Odometry modality to show the evolution of the emergent concept of continual learning (refer Section 6.1.1)—a brief description of the results from other modalities (i.e., control $S-P$ and control $S-V$) provided.

- **Odometry**: An initial filter (i.e., unmotivated Kalman filter) applied to the Scenario I perimeter monitoring (PM) data of odometry (refer Fig. 6.9) and obtained generalized errors (GEs) as output. By applying the GNG algorithm on the GEs (i.e.,GE0 and GE1) , discrete cluster space generated as shown in Fig. 6.12a and Fig. 6.12b respectively. The same colored nodes in plots Fig. 6.12 represent the mapping of GE0 and GE1 space found by the co-occurrence matrix.

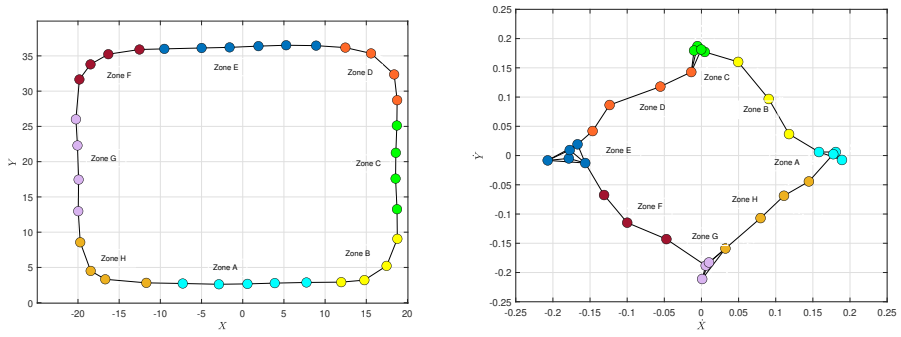
Each type of dynamics (i.e., horizontal, vertical, and curve motion) and location co-ordinates (i.e., lower, upper, right, and left) of the ego-things PM task trajectory (refer Fig. 6.9) enable a different subset of nodes in GE0 and GE1 cluster space. For example, Zone A (horizontal lower) in GE0 space (refer Fig. 6.12a) maps to Zone A (the cloud of cyan coloured nodes) in GE1 space (refer Fig. 6.12b). It is evident from the plots that the odometry modality extracts spatial features to detect a spatial abnormality.

A filter $A1$ collectively learned from the information acquired by GEs cluster spaces and co-occurrence matrix of Scenario I perimeter monitoring (PM) data of odometry can predict the future nodes enable in GE0 and GE1 space (refer Section 6.1.1) and their correlation. This filter $A1$ tested with Scenario II, ES1 dataset (refer Fig. 6.11a) where the ego-thing pass through a different dynamics (i.e., emergency stop operation). Here the prediction discrete nodes (*letters*) mismatch with the nodes(letters) enabled by the observed sensory data sequence everywhere except the interval where the emergency stop operation performed and is shown in Fig. 6.13a. The projected segment and the nodes in *red* color shows the presence of an anomaly.

Whenever the ego-thing passes through new experience (i.e., detected anomalies), it will automatically execute a new filter model learning from the new experience dataset. A filter called $A2$ learned from this data can represent similar scenarios in the future with embedded knowledge. In the next step, the filter ($A2$) tested with another dataset of Scenario II, ES2 (refer Fig. 6.11b) where the pedestrian appears at two spatial locations of the vehicle maneuvering trajectory. The estimated anomaly is shown as the projected segment and nodes in *red* color in Fig. 6.13b. The additionally enabled nodes are only in one spatial location (i.e., on the right side) even though the emergency stop operation performed twice. It means that the filter $A2$ was well able to encode the first emergency stop with the embedded knowledge as it happened in the same spatial location of the data used to learn filter $A2$. But the second emergency stop operation performed in a different location, and $A2$ was unable to represent this situation

and generated an anomaly.

From this anomaly data, the ego-thing will learn a new filter(i.e.,$A3$) that can embed this new experience's knowledge to make inference in the future when the ego-thing pass through a similar experience. If we analysed the plots, Fig. 6.12b, Fig. 6.13 (a) and Fig. 6.13 (b) together, the evolution of emergent concept is self explanatory. Whenever the system endures new experiences, automatically learn new filters to represent similar future experiences of ego-things (by the knowledge encoded in the learned filters). Consequently, contextual awareness and the collective decision-making process of the system increases.



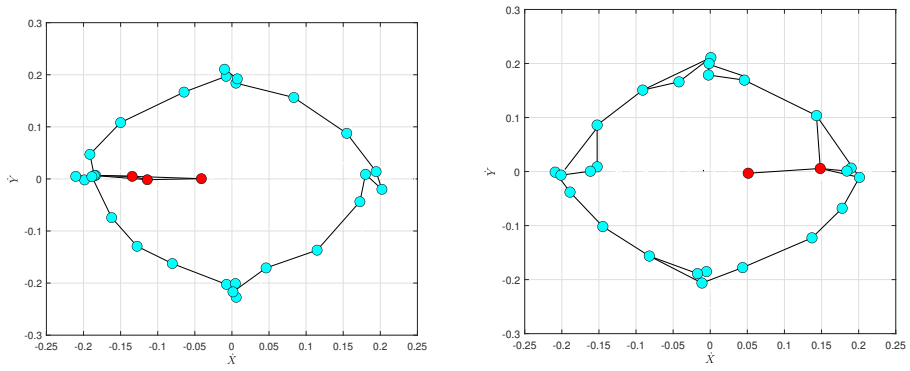
(a) GE0 space.

(b) GE1 space.

Figure 6.12: Clustering of GEs of odometry X-Y (training data). Nodes indicate the cluster centers of associated data points.

– **Control *S-P*** : This modality considers the proprioceptive sensory data of the control steering-power $(S-P)$ combination.The generalized errors discrete spaces of produced from the PM task (Scenario I) control S-P plotted in Fig. 6.14. The clustering of the GE0 shown in Fig. 6.14a and Fig. 6.14b is the GE1 space (i.e. $\dot{S}-\dot{P}$ discrete space).

The nodes marked as Zone B in Fig. 6.14a and Fig. 6.14b shows the mapping between GE0 and GE1 spaces captured by the co-occurrence matrix. In GE0 space (refer Fig. 6.14a), the steering angle values are either zero or near to zero for the linear movement of the ego-thing, and the values become more negative during

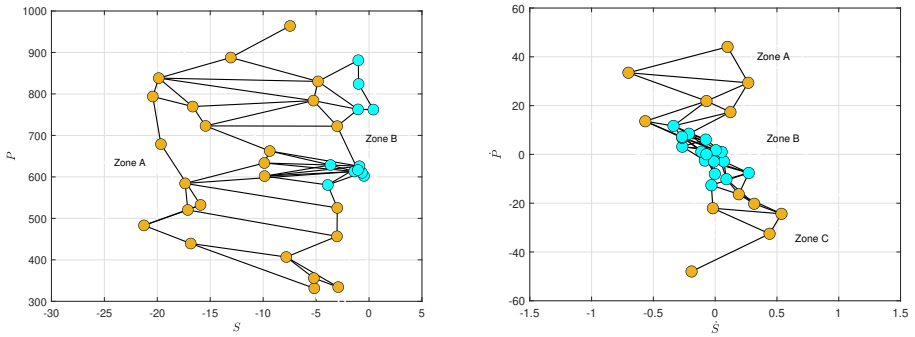


(a) (b)

Figure 6.13: Emergent concept of odometry: GE1 space (a) Test data 1 (ES1); (b) Test data 2 (ES2). The projected segments and the nodes in red colors indicate the presence of abnormality.

the movement in curves(the considered datasets only consist the left side curved movements). Simultaneously, the power values are almost stable during rectilinear movements, and in curves, it acquires different values. Similarly, in the GE1 space (refer Fig. 6.14b) Zone B represents the linear movements, Zone A and Zone C correspond to the nodes activated during the vehicles' curved motion. Contrary to odometry modality, S-P modality is good for differentiating the types of ego-things different dynamics. The learned filter from the generalized errors detects abnormalities when the prediction varies from the ground truth observed data.

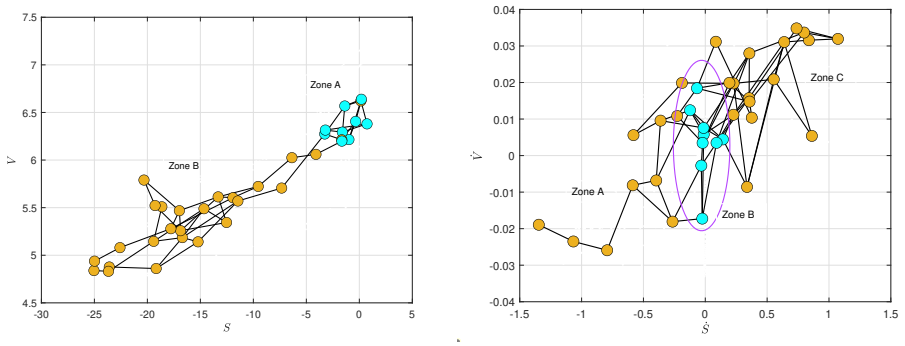
We have analyzed the GE spaces of control *S-P* training and test datasets in line with odometry. When the learned filter from generalized errors of training dataset tested with ES1 and ES2 task of Scenario II, few additional nodes activated to represent the emergency brake operation abnormality. Each of the abnormality is considered as a new feature to learn new filters. Contrary to odometry, the concept learned with control *S-P* able to detect and differentiate the anomaly during either the ego-things are in linear motion or the curved trajectory. The spatial location is



(a) GE0 space (b) GE1 space

Figure 6.14: Clustering of GEs: Control S-P (training data). Nodes indicate the cluster centres of associated data points

not significant in this case. This emergent concept learned from the proprioceptive control modality enriches the self-awareness of each ego-things in the network.



(a) GE0 space (b) GE1 space

Figure 6.15: Clustering of GEs: Control S-V (training data). Nodes indicate the cluster centers of associated data points.

– **Control *S-V*** : The plots of discrete space for control S-V modality are shown in Fig. 6.15. Zone A in GE0 space (refer Fig. 6.15a) represents the rectilinear movement of the ego-thing, and it enables the nodes located in Zone B of GE1 space, as shown in the Fig. 6.15b. Similarly, Zone B in GE0 space activates either Zone A or Zone C in the GE1 space. During linear movement, the

steering acquired zero or nearby values, but the velocity would be maximum. During curves, steering values can be more positive or more negative. In our considered scenarios to collect datasets, the vehicles perform only curve to the left side, so that steering values are more negative. This modality helps to understand the different movement patterns of the ego-things and this will enrich the SA of each ego-things .

The concept learned for S-V modality shown similar results of S-P modality except for the difference in the collective behavior of the data variables considered.

The continual learning of filters from ego-things new experiences are self-explainable in this sense. The peculiar features will be encoded inside the filter learned from different experiences of ego-things. The filter models learn and update incrementally whenever the system passes through new experiences, as shown in Fig. 6.5. In this way, the agents are more intelligent; they have the functionality of detecting abnormality and describing it at different abstraction levels.

6.3.2 Phase 2: Anomaly detection by D-MJPF

In this part, we have applied D-MJPF on the CDBN models (Filter(s) B in Fig. 6.5) learned from the data sequences by considering three different modalities. Inside each ego-thing, three models learned in total from the data of perimeter monitoring task performed by two vehicles by considering three low dimensional data combinations, i.e., X-Y position odometry data, steering-power (S-P), and steering-velocity (S-V).

To test the models' efficiency, we have used the ES1 dataset of the aforementioned variable combinations of Scenario II. The models were able to detect the vehicles' emergency brake's abnormal behavior when a pedestrian appears in front of the header vehicle. Fig. 6.16, Fig. 6.18 and Fig. 6.17 shows the abnormality plots of odometry, control $S-P$ and control $S-V$ respectively for iCab 1 and iCab 2 vehicles. The region inside the dotted rectangular box represented the interval when vehicles performed emergency brake operation. The abnormality

metric used was the *innovation* of the D-MJPF. i.e., the difference between the predicted states and the ground truth observations (refer Eq. 6.12). As shown in Fig. 6.16, Fig. 6.18 and Fig. 6.17, there is a significant rise in the *innovation* measurements during the intervals when the emergency brake operation executes.

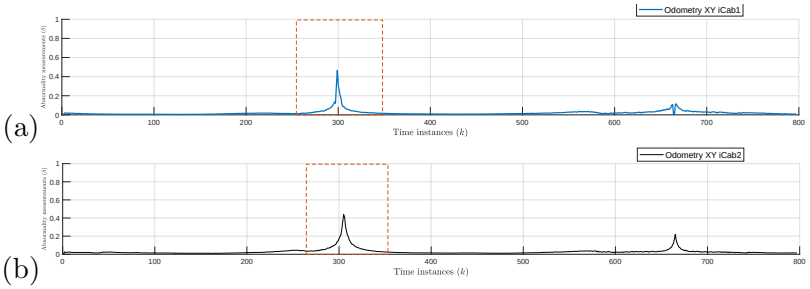


Figure 6.16: Abnormality measurements for odometry (a) $iCab1$, (b) $iCab2$

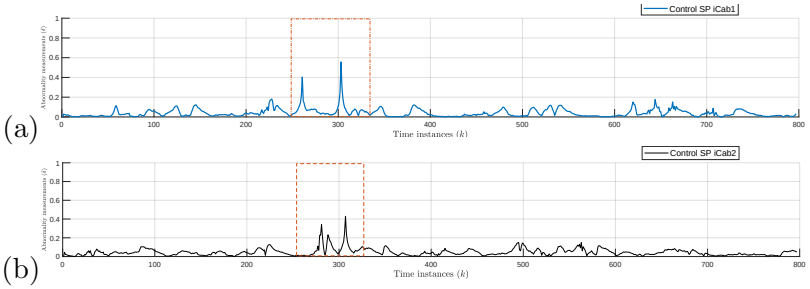


Figure 6.17: Abnormality measurements for control (SP): (a) $iCab1$, (b) $iCab2$

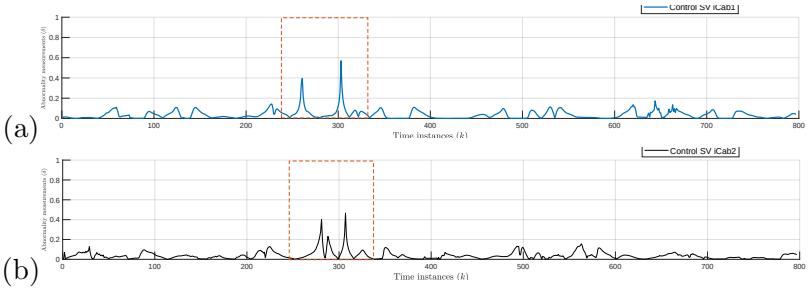


Figure 6.18: Abnormality measurements for control (SV): (a) $iCab1$, (b) $iCab2$

The data-driven models can not only provide a global estimation of anomalies based on the whole set of multidimensional generalized variables used in the models but also provide an insight anomaly related to single specific components of the model. For example, the model

learned from *S-P* data sequences was able to estimate the behavior of only steering (S) or only velocity (V) of the vehicle efficiently. It is an additional explainability feature of the model.

6.3.3 Evaluation of model performance after the channel effects

Inside each vehicle, we have three different CDBN models (represented as three different colored blocks in Fig. 6.5), and the models inside each ego-thing are the same. In Phase I and Phase II of DBN model testing, we assumed all the ground truth observations are available to all the ego-things without data packet loss and delay. This section presents the performance comparison results of CDBN models under the channels influence on transmitted data.

As described before, the Opportunistic Network Environment (ONE) simulator here used to measure the channel effects over the data transmitting data. A network can be affected by different types of delays, such as a propagation delay, transmission delay, queuing delay, and processing delay [59]. However, in this work, we considered the propagation delay, and the packets arrive with a considerable delay and are considered equivalent to lost packets.

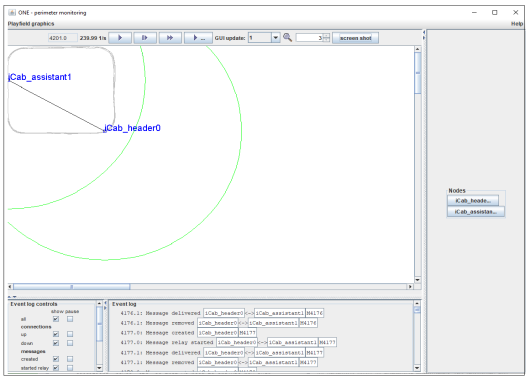


Figure 6.19: GUI of ONE simulator

In ONE simulator, there are six routing protocols included (Direct Delivery (DD), First Contact (FC), Spray-and-Wait, PRoPHET, MaxProp, and Epidemic),and set the movement model as *MapBasedMovement* [55]. However, we have chosen Direct Delivery (DD) as the

number of dynamic objects is limited to two in this work. The real trajectory data of the PM task (Scenario I) described in Section 6.2 is inserted in the simulator as the Well Known Text (WKT) file format and created two dynamic nodes that represent the header(iCab1) and assistant(iCab2) vehicles. Fig. 6.19 shows the simulator environment.

The parameters used in ONE simulator are summarized in Table 6.1. Both of the protocol have some features and, at the same time, some limitations. For example, IEEE 802.15.4 protocol allows low power transmission, but we need to compromise with the low data rate, which leads to more packet loss. On the contrary, the IEEE 802.11p protocol supports a high data rate, but the transmission power is comparatively high. Different tests performed with the aforementioned

Table 6.1: Simulation parameters for ONE simulator by considering protocol IEEE 802.15.4 and IEEE 802.11p

	Parameter	**IEEE 802.11p**	**IEEE 802.15.4**
1	Frequency band	5.9GHz (Licenced)	2.4GHz (Unlicensed)
2	Data rate	18 Mb/s (16 QAM) 27Mb/s (64 QAM)	50Kbps
3	Transmission power	2W(33dBm)	3mW(4.77dBm)
4	Receiver sensitivity (dBm)	-73,-68	-85
5	Transmission range	100m	100m
6	K-values	0,3	0,3
7	Data packet size	48 Bytes	29 Bytes

protocols with different data rates, transmission power, and K-factor, as shown in Table 6.1. High K-factor values refer to the rural environment with the presence of obstacles, buildings, etc. that have a lower impact on the CDBN model performance. The K-factor value of zero represents an environment where no line of sight (NLOS) components are available, and such condition negatively affects the model performance. The receiver sensitivity column shows the minimum values of the Signal-to-Noise Ratio (SNR) at the receiver to guarantee successful data reception [18]. The data delivery probability between the sender vehicle and receiver vehicle has been estimated and summarised in Table 6.2. As expected, the probability values are low where the NLOS component presents, i.e., K=0 and increased for the LOS scenario, i.e.,

Table 6.2: Data delivery probability over two different protocols,data rates and K-values.

K-values	**IEEE 802.15.4, 50Kbps**	**IEEE 802.11p, 18Mb/s**	**IEEE 802.11p, 27Mb/s**
0	0.9796	0.9973	0.9934
3	0.9960	0.9996	0.9990

K=3 (rural environment). When more data packets lose, the CDBN model performance degrades in the estimation of future sates of the vehicles, and as a result, abnormality estimation gets affected.

The abnormality estimated in Section 6.3.2 by the D-MJPF in ideal condition compared with the estimated anomaly in the presence of channel effects. For the evaluation of models, we have used different evaluation metrics such as MSE, Accuracy and F1 score(refer Sec. 6.1.2).

We presented here the results for the header vehicle (iCab1) only. The CDBN models of the header vehicle(iCab1) estimated anomaly by own multi sensory observed data. Simultaneously, the same data transmitted to the assistant vehicle (iCab2) over the wireless channel. The CDBN models inside the assistant vehicle estimate the header vehicle's abnormality along with its own abnormality. We have conducted the simulation by considering two protocols with three different data rates, environmental conditions (K-factor values), and three different modalities. As explained before, the three different modalities extract different feature of the ego-things to enrich the self/collective awareness of the ego-things network.

Modality	XY			SP			SV		
Environmental factor	No loss	K=3	K=0	No loss	K=3	K=0	No loss	K=3	K=0
MSE	0	0.0047	0.0204	0	0.0048	0.0193	0	0.0047	0.0204
Accuracy	0.9975	0.9876	0.9777	0.9950	0.9851	0.9752	0.9913	0.9814	0.9739
F1 score	0.9667	0.8529	0.7632	0.9355	0.8286	0.7436	0.8852	0.7826	0.7273

Figure 6.20: Model performance evaluation: IEEE 802.15.4

The estimated results (MSE, Accuracy and F1 score) are summarised in Table 6.20, Table 6.21, and Table 6.22. The models performed

Modality	XY			SP			SV		
Environmental factor	No loss	K=3	K=0	No loss	K=3	K=0	No loss	K=3	K=0
MSE	0	0.0012	0.0036	0	0.0011	0.0033	0	0.0012	0.0034
Accuracy	0.9975	0.9963	0.9950	0.9950	0.9926	0.9913	0.9913	0.9888	0.9876
F1 score	0.9667	0.9508	0.9355	0.9355	0.9063	0.8923	0.8852	0.8571	0.8438

Figure 6.21: Model performance evaluation: IEEE 802.11p, data rate: 18Mb/s.

Modality	XY			SP			SV		
Environmental factor	No loss	K=3	K=0	No loss	K=3	K=0	No loss	K=3	K=0
MSE	0	0.0024	0.0072	0	0.0024	0.0067	0	0.0023	0.0070
Accuracy	0.9975	0.9950	0.9901	0.9950	0.9901	0.9888	0.9913	0.9864	0.9839
F1 score	0.9667	0.9355	0.8788	0.9355	0.8788	0.8657	0.8852	0.8308	0.8060

Figure 6.22: Model performance evaluation: IEEE 802.11p, data rate: 27Mb/s .

least (highest MSE value) when used the IEEE 802.15.4 protocol (refer Table 6.20) standard and K-factor value is zero.

However, the MSE metric is not enough to perform an in-depth analysis of the model performance as it doesn't take into account accuracy, precision, etc. So that we further estimated the accuracy and F1 score for better studying and analyzing the model performance. Accuracy and F1 score were high when used IEEE 802.11p with a data rate of 18 Mb/s as in Table 6.21, and MSE values were least in this case as expected. This is considered as the best performance under the channels' influence. Table 6.22 summarises the model's performance when used IEEE 802.11p with a 27Mb/s data rate. The Accuracy and F1 score were better than IEEE 802.15.4 and worse than IEEE 802.11p, 18Mb/s data rate.

In summary, data rates, transmission power, received sensitivity, environmental conditions, etc. plays a role in the model performance. We need to carefully set the parameters and choose the appropriate protocol by studying the application area and the resources available. In this work, the payload size was not so big, so that the model performances didn't degrade too much. Once the payload size goes high, it affects the model performance. In the future, we will extend the work

with a bigger payload size and also include more parameters.

6.4 Chapter summary

This article presented a method to develop multimodal collective awareness for networked IoT nodes performing joint tasks. The IoT nodes in his work are autonomous vehicles, and each of the vehicles is assumed to be having machine learning capabilities. The CDBN models learned from exteroceptive and proprioceptive sensory data have the functionality to extract unique features of the system related to self and contextual awareness and detect abnormalities happening anywhere in the networked ego-things. The CDBN models are data-driven and capable of detecting abnormalities at different abstraction levels. The distributed state estimation performed by MJPF is associated with each CDBN model. The models inside each agent can synchronously estimate the possible abnormalities around any of the agents in the network. Moreover, the models can describe anomaly related to single specific components of the vector used for model learning; this is an additional explainability feature of the models.

In the offline training phase, the multisensory data collected while the agents performing a joint task used to learn the CDBN models. In the online test phase, the model's fitness tested with the datasets from a different joint task than used in the training phase. The presented results at different abstraction levels provide evidence for our method's efficiency in detecting abnormal situations in the networked agents. Moreover, we have analyzed the effects of wireless communication channels on the model performance by considering different protocols and channel conditions and finally compared the obtained results by different performance evaluation metrics.

Chapter 7

Interpretable Machine learning models for abnormality detection in Ego-things network

In recent days, it is becoming important to ensure that the outcomes of signal processing methods based on machine learning (ML) data-driven models can provide predictions that can be interpreted. The interpretability of ML models can be defined as the capability to understand the reasons that contributed to generate a given outcome in a complex autonomous or semi-autonomous system. The necessity of interpretability is often related to the evaluation of performances in complex systems as well as to the acceptance of automatization processes of agents where critical high-risk decisions have to be taken. This chapter concentrates on a functionality of such systems, i.e., abnormality detection, and on the choice of a model representation modality based on a data-driven machine learning (ML) technique such that the outcomes become interpretable. The proposed approach assumes that the data-driven models to be chosen should support emergent SA of the agents at multiple abstraction levels. It is demonstrated that the capability of incrementally update learned representation models based on progressive experiences of the agent is strictly related to interpretabil-

ity capability.
As a case study, abnormality detection is analyzed as a basic feature of the CA of a network of vehicles performing cooperative behaviors. Each of the vehicles is considered as an example of an Internet of Things (IoT) node, so providing results that can be generalized to an IoT framework where agents have different sensors, actuators, and tasks to be accomplished.
The capability of a model to allow evaluation of abnormalities at different levels of abstraction in the learned models is addressed as a key aspect for interpretability.

The main contributions of this chapter can be summarized as follows:

- A novel method is proposed to learn data-driven ML models representing self-awareness (SA) and interactive collective awareness (CA) of agents' network. The learned model is interpretable, i.e., the model is self explainable about the results it produces and the decisions made in different situations. For the inferences, a MJPF based on generative DBN models is used and extended to become able to detect local and global abnormalities.
- The system has the capability of emergent incremental learning of new models when the agents encounter new experiences, i.e., when the model detects abnormal situations. The different abstraction level results of abnormalities generated by the models are presented and compared. In this work, interpretability is defined as the capability to use anomaly data to modify the existing model used to obtain the anomaly itself.

7.1 Machine learning models and interpretability: Design and Implementation

The term ego-thing used in this chapter refers to intelligent agents that can perceive their internal and external parameters and adapt themselves when they face abnormal situations [52]. The below sections describe the model learning, testing, and interpretability by considering a two-agent network.

The generative DBN (GDBN) model learned from low-dimensional data mainly focuses on detecting local and global abnormalities in the agents' network. The model we propose is generative, and it is not only capable of detecting anomalies at multiple abstraction levels but also uses the generalized errors (GEs) of produced anomaly data to incrementally learn new models. The incrementally learned models allow the interpretation of detected anomalies at different abstraction levels. At the same time, learning is a process of finding a new DBN that minimizes the presence of GEs in a given sequence.

In this work, GEs are defined as the mismatch between the Bayesian predictions and observed evidence, i.e., the states and the deviations of states' (i.e., higher-order derivatives) together. The higher-order derivatives are limited to first-order, and the GEs have been detected through the anomaly detection process.

In the model training phase, the GEs produced from the sensory data have been used to learn the model. In the test phase, when the model produces the anomaly by estimating the probabilistic distance between prediction and evidence, new models will be learned incrementally by exploiting the GEs produced from the anomaly data that correct/minimize those errors. In other words, the model not only detects anomalies at multiple abstraction levels but also uses the produced GEs (of an anomaly) to correct or learn a new model.

7.1.1 Offline phase: Model learning

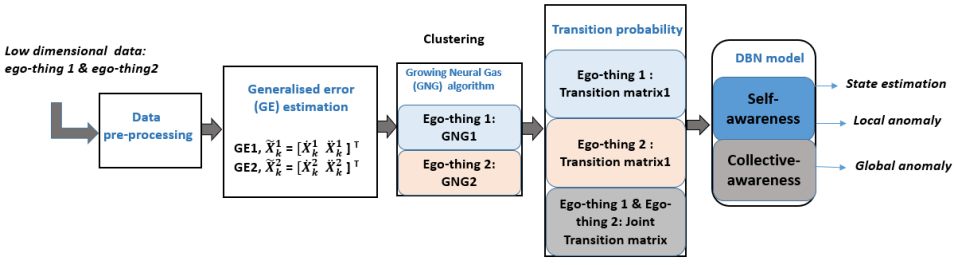


Figure 7.1: DBN model learning process.

This section detailed the processes involved in building a model that can provide the functionality of awareness and interaction in a network

of agents. The block diagram of the model learning process is shown in Fig. 7.1. In this work, we limited the number of agents to two and assumed all the sensory data collected by the ego-things have been available for both the agents in the offline and online phases. The communication part is not considered in this chapter.

The data-driven model has different abstraction levels, such as 1) Continuous and 2) Discrete. The continuous level focuses on SA functionality to detect the local anomaly. This level of the model directly related to sensory observations at sub-symbolic level. On the other hand, the discrete level learned at a symbolic level from both agents' joint data is dedicated to joint anomaly detection, i.e., a global anomaly. The peculiarity of the learned model is that it can represent treating the ego-things separately and jointly. Such SA and joint awareness functionality can be used to predict the agents' future states and detect local and global anomalies.

Estimation of generalized errors (GEs)

Firstly the data sequence collected from different agents has been synchronized to match their time stamps. This work mainly considered the proprioceptive control data of rotor velocity (V) to build the model. The chosen sensory data to develop the model is low dimensional, i.e., 1D vector for one agent. Let Z_k^{en} be the measurements from the ego-thing en at the time instant k and X_k^{en} be the associated latent state variable. The measured observations of ego-thing n can be mapped to the latent states by the following observation model:

$$Z_k^{en} = f(X_k^{en}) + \boldsymbol{\delta}_k \tag{7.1}$$

where $\boldsymbol{\delta}_k$ represents the vector composed of measurement noise at a time step k. and the function $f()$ is assumed to be linear.

The GEs can be produced by applying Unmotivated Kalman Filter (UKF) [27] on the considered data sequence. This filter assumes that the state vector at time instant $k+1$ will be the same as the previous time instant k. When the observed data sequences violate this rule, GEs will be produced, and it is equivalent to derivatives of the data.

In this work, we have limited the GEs to first-order derivatives. The GEs related to ego-thing en can be written as:

$$\tilde{\mathbf{X}}_k^{en} = [X_k \;\; \dot{X}_k \;\;]^\intercal, \tag{7.2}$$

As shown in Eq. 7.2, we have used states and their first-order derivatives as the GEs. The generalized errors are used in the next section to learn the discrete state variables of the model.

Transition probability from discrete vocabulary

The model's discrete abstraction level has been built from the discrete vocabulary variables and different transition matrices. Firstly, the clustering operation is performed on the GE of each ego-thing separately. In this work, we have used GNG algorithm [39] for clustering the GEs. The reason why we choose this algorithm over other clustering algorithms is described in section 4.1.1. GNG is an unsupervised incremental neural network to learns typologies and it does not require users to define the number of nodes /clusters beforehand. This dynamic property of GNG is an advantage over other clustering algorithms for using it in many applications. By considering the nature of the dataset used in this work, we have selected the GNG algorithm by taking into account its advantages over other clustering algorithms.

The GEs that belong to each ego-thing is separately clustered by the unsupervised clustering algorithm of GNG algorithm [39]. Each cluster is represented by its centroid/node, a 2D vector, and the centroid is a mean value of all the GEs belonging to that particular cluster.

The input of each GNG is a 2D vector, i.e., generalized errors of states and first-order derivatives. Therefore, one GNG is dedicated to each ego-thing. For example, the input vectors to the GNGs belong to the ego-things are in the form as below.

$$Ego\ thing\ 1,\;\; X_k^1 = [v_1 \;\; \dot{v}_1]^\intercal \tag{7.3}$$

$$Ego\ thing\ 2,\;\; X_k^2 = [v_2 \;\; \dot{v}_2]^\intercal \tag{7.4}$$

For instance, the group of nodes created by a GNG of each ego-thing can be written as:

$$\boldsymbol{S}^{e1} = \{P_1, P_2, \ldots, P_m\} \tag{7.5}$$

$$\boldsymbol{S}^{e2} = \{Q_1, Q_2, \ldots, Q_m\} \tag{7.6}$$

where m represents the maximum number of nodes/clusters produced by the GNG.

$$W^c = [P_a \;\; Q_b]^T \tag{7.7}$$

where P_a represents the a^{th} element of the group of nodes produced by GNG1 belongs to ego-thing 1. Likewise, Q_b represents the b^{th} element of the list of nodes generated by GNG2 (i.e., the GNG belongs to ego-thing 2). The next step is to estimate the transition probability matrices for each ego-thing separately and also joint transition links. The example of the transition probability matrix belong to ego-thing 1 can be written as below:

$$T_{e1} = \begin{pmatrix} P_{11} & P_{12} & \ldots & P_{1n} \\ \vdots & \ddots & & \\ P_{n1} & P_{n2} & \ldots & P_{nn} \end{pmatrix} \tag{7.8}$$

where n represents the total number of nodes produced by the GNG, each element in the matrix represents the probability of transition between the discrete random variables (belong to ego-thing 1).

Then estimate the joint transition matrix from the list of words (refer eq. 7.7). This matrix represents the joint transition probability between the two ego- things as below.

$$T_{joint} = \begin{pmatrix} W_{11} & W_{12} & \ldots & W_{1m} \\ \vdots & \ddots & & \\ W_{m1} & W_{m2} & \ldots & W_{mm} \end{pmatrix} \tag{7.9}$$

where m represents the total number of joint vocabulary variables (i.e., *words*) and each element represents the transition probability between

those variables.

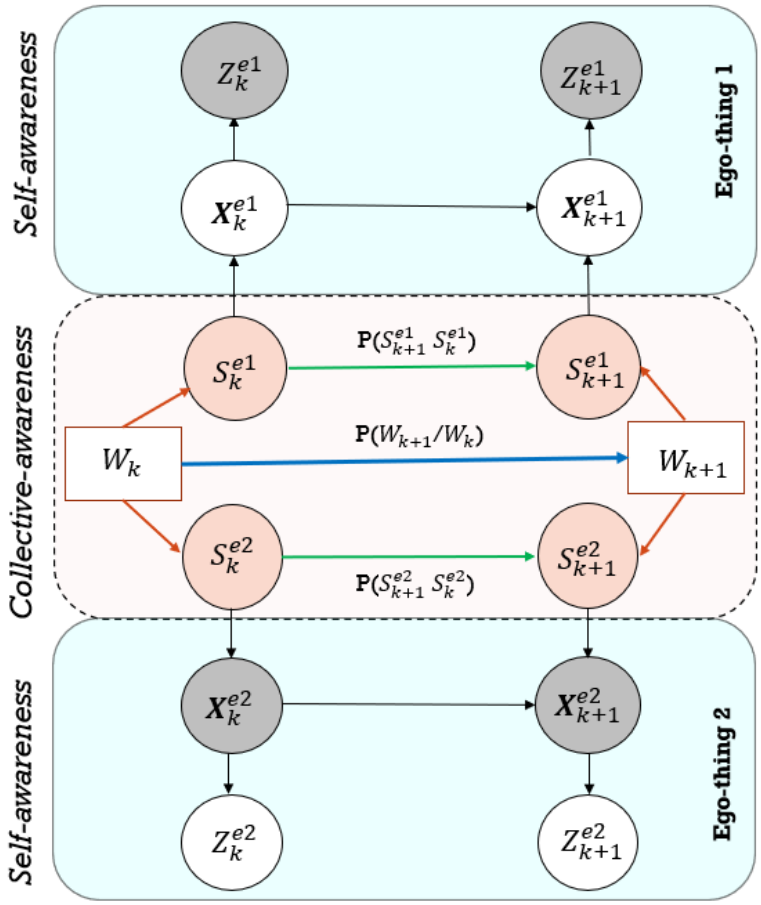


Figure 7.2: GDBN model for a two ego-thing network: The cyan and orange shaded area of the model represents SA and CA respectively. Horizontal and vertical arrows represent the probabilistic connection between the variables and different abstraction levels.

7.1.2 Online phase: Model testing

In this phase, a D-MJPF is applied to the learned generative DBN model to make inferences on the observed data. The block diagram representation of the model testing and continual incremental learning of new models shown in Fig. 7.3. The model can detect global as well as local anomalies arise in the agents' network. Global anomaly means the anomaly happens anywhere in the network, and local anomaly

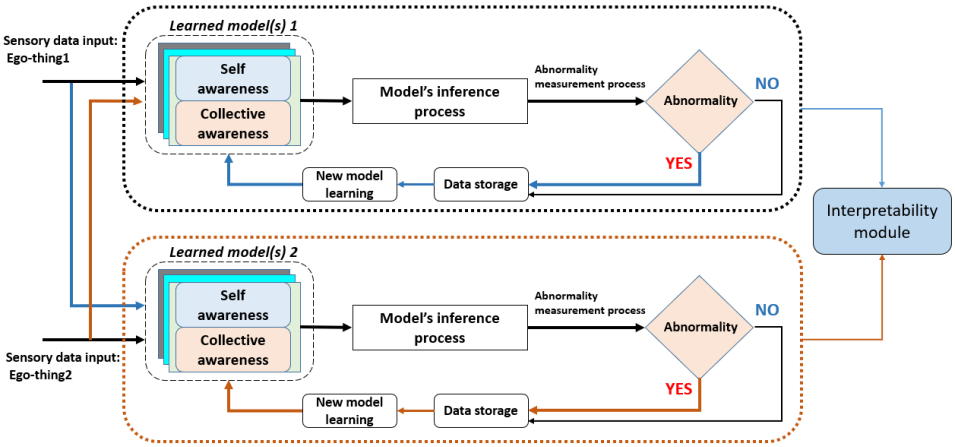


Figure 7.3: General block diagram of model testing and continual learning.

focuses on the anomaly around a particular agent.

Anomaly detection: D-MJPF

The MJPF [32] is a hybrid filter; KFs at the continuous level and PFs at the discrete level to make inferences. The detailed description of the MJPF is described in [52, 14]. However, in this chapter, we have used a modified version of it by adding the functionality to detect discrete level global anomaly along with the local anomaly detection capability.

The inference starts with an initial observed random data point. It then estimates the discrete node by calculating and selecting the cluster nodes with a minimum distance with the ground truth observation. The selection of control vector for the future state prediction by the continuous level part (i.e., Kalman filter) is influenced by the probability of discrete level variable estimated before. The prediction at the continuous level is made separately for both agents, which is part of the model's SA functionality. The particle filter at the discrete level makes the inference of joint state prediction and global abnormality estimation. Moreover, the system can continually learn new models whenever the ego-things passes through new experiences.

– Local anomaly detection: Self-awareness
  The continuous level of the learned DBN model (refer to Fig. 7.2)

is dedicated to predicting future states of ego-things separately and detecting anomalies around a specific ego-thing. To estimate the anomaly at the continuous level, we have used the *innovation* term of the D-MJPF and can be estimated as below:

$$\theta_{k,e1} = Z_k^{(e1)} - HX_k^{(e1)} \tag{7.10}$$

$$\theta_{k,e2} = Z_k^{(e2)} - HX_k^{(e2)} \tag{7.11}$$

where θ_k represents the innovation terms, Z_k term represents the ground truth sensory observation, and X_k is the predicted states. The GEs produced (refer Eq. 7.10 and Eq. 7.11) from the anomaly detection process has been used to incrementally learn new models to represent the situation.

Once detected anomaly at continuous level (i.e., local abnormality), the GEs produced from the anomaly signal has been used to correct the model incrementally. Here the GEs are the predicted states and the anomaly measurements (i.e., the probabilistic distance between predicted states and observed evidence). The new model has to capture the new situation that produced anomalies and predict the states better when similar situations will occur.

The GEs signal of the anomaly (i.e., a mismatch between model predictions and updates) has been firstly clustered by applying the GNG algorithm. These clusters generated from the GEs will mainly differ from previously generates clusters on those intervals where the anomaly occurred. Then the information extracted by the clustering is used to learn vocabulary and transition probability matrix. The transition probability (shown as green arrows in Fig. 7.2), which influences the model's continuous level, will be improved to predict the situation well and consequently to produce low or null GEs.

– Global anomaly detection: Collective awareness

The DBN models' discrete level (shaded in orange color) shown in Fig.7.2 is dedicated to detecting the global anomaly, i.e., the anomaly happening anywhere in the network. In co-operative task scenarios, this metric can detect the anomaly happening

around other agents in the network. The metric used to estimate anomaly is Kullback–Leibler divergence [72]. The KL divergence is a measure of how a probability distribution differs from another probability distribution. This work has used the metric to estimate the difference between the predicted discrete level variables distribution and the discrete state's distribution estimated from the observed sensory variables. All the variables considered here in the discrete level are estimated jointly by considering both ego-things. The below equation can estimate the KL distance:

$$D_{\mathrm{KL}}(\lambda \parallel \pi) = \sum_{x \in \mathcal{X}} \pi(x) \log \left(\frac{\lambda(x)}{\pi(x)} \right) \tag{7.12}$$

where π is the joint distribution of predicted discrete random variables and λ is the joint distribution of observed discrete state variables.

Once detected global abnormality, i.e., anomaly happens anywhere in the ego-things network, firstly list out those ego-things that encountered abnormal situations locally. Then the GEs of abnormality signals detected by the ego-things continuous level will be clustered separately. The clustered information will be used to update the corresponding discrete vocabulary and transition probability matrices. The very next step is to update the joint vocabulary (*words*) and joint transition probability matrix (represented by the *green* arrow in Fig. 7.2). Therefore, if a similar joint event occurs in the future, the model can represent the situation to make appropriate decisions.

Continual model learning and interpretability

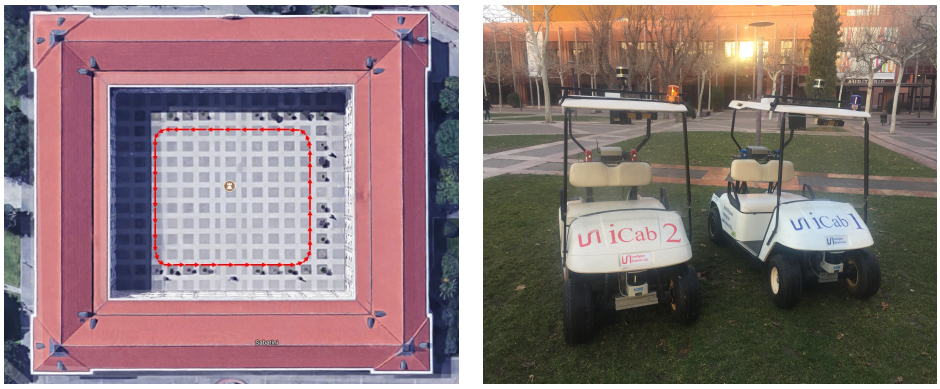
Model learning is a continuous process whenever the existing models cannot represent the ego-things current experience, i.e., when the model detects an abnormality. The developed ML model's interpretability focuses on the model's intermediate results, such as discrete level vocabulary and transition probability matrices. Section 7.3 elaborated the concept of interpretability with results from different abstraction

levels of the model.

Once the model detected abnormality, the continual model learning phase will be enabled. The GEs produced from the detected anomaly signal will be initially clustered. Then a new vocabulary and transition probability matrices will be learned of the new model. The correction of the existing model will be performed at the model's different abstraction levels based on the anomaly signal.

7.2 Experimental study

This section explains the case study and the datasets used to validate the proposed methodology. Two intelligent autonomous vehicles named *iCab* (Intelligent Campus Automobile) having the same setup [42] used in this work and shown in Fig. 7.4b.



(a) Testing environment. (b) iCab vehicles.

Figure 7.4: The environment and the vehicles used for the experiments.

Each vehicle is equipped with sensors, such as one lidar, a stereo camera, laser rangefinder, and encoders. This work focused on the low-dimensional control data, i.e., rotor velocity (v) of the vehicles. The collected data from both vehicles are synchronized to align their timestamps. The two iCab vehicles perform joint navigation tasks in the rectangular trajectory shown in Fig. 7.4a by keeping their position one after the other with a minimum distance among them. The vehicle navigates in the front called *leader* (*iCab1*) and the one follows is the *follower* (*iCab2*).

7.2.1 Training datasets

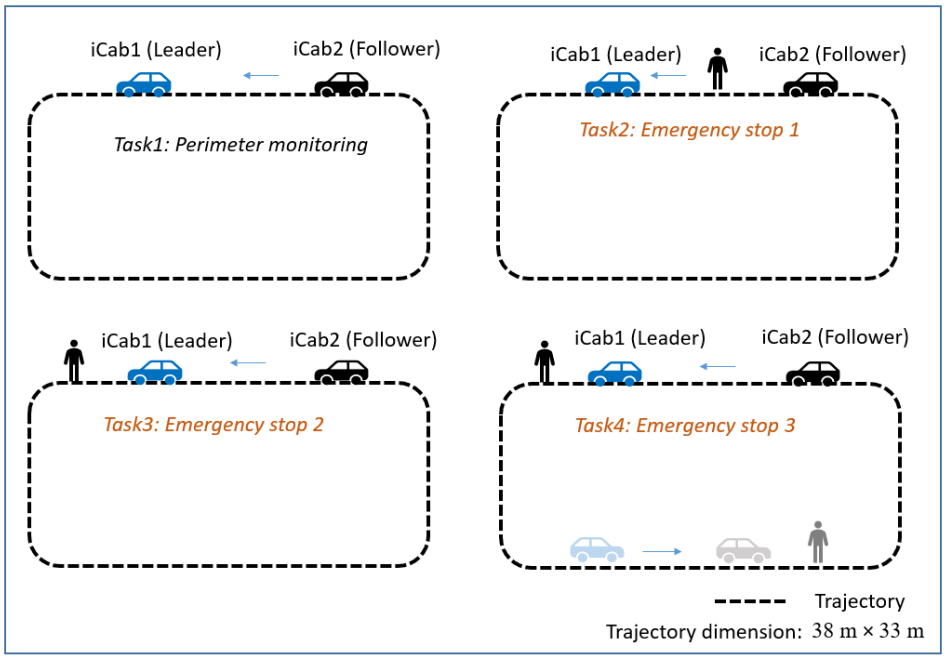


Figure 7.5: Cooperative driving tasks: The different cooperative driving scenarios considered. The dataset from Task 1 was used in the training phase to learn the model, and the remaining tasks (Task 2 to Task 4) were used in the test phase to check the model's fitness in detecting local and global abnormality.

Perimeter monitoring: The vehicles jointly perform platooning operation in a closed environment, as shown in Fig. 7.5: Task 1. The navigation operation was performed four times, one after the other, and collected the data. The *follower* vehicle (*iCab2*) mimics the actions of the *leader* (*iCab1*) vehicle.

7.2.2 Test datasets

The example of rotor velocity test data for *iCab1* and *iCab2* when performs joint navigation operation of Task 3 plotted in Fig. 7.6. During the emergency brake operation (when the vehicle encounter a pedestrian), the velocity values are reduced compared to the training data sequence and are marked by the red rectangular box. The rotor

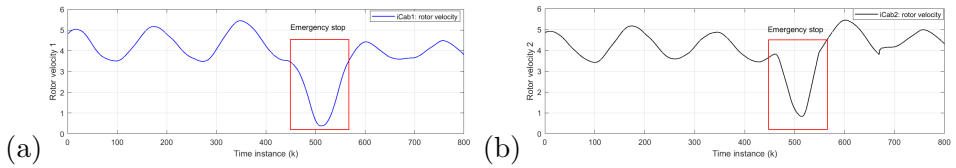


Figure 7.6: Test data: Emergency stop (a) *iCab1* rotor velocity (b) *iCab2* rotor velocity.

velocity data (V) of both vehicles while performing the below tasks have been collected and used in the learned model's test phase.

1. Emergency stop1: While both vehicles jointly navigate a rectangular trajectory, a random pedestrian appears in front of the follower vehicle (*iCab2*), and it performs an emergency brake operation. Since the leader vehicle not encountering any obstacles, it continues the navigation task. The follower vehicles continue the navigation once the pedestrian crosses the danger zone. Fig. 7.5: Task2 depicts this scenario of joint navigation.
2. Emergency stop 2: The task is similar to the previous one (Emergency stop1), except the pedestrian appears in front of the leader vehicle (*iCab1*). As soon the leader detects the presence of a pedestrian, it performs an emergency brake operation. Subsequently the follower (*iCab2*) vehicle also performs emergency brake (refer Fig. 7.5: Task3). Once the pedestrian crosses, both the vehicles (*iCab1* and *iCab2*) continue the joint perimeter monitoring task.
3. Emergency stop 3: The task is similar to previous scenarios, but in this case, the pedestrian appears in two locations in the trajectory (refer to Fig. 7.5: Task 4).

7.3 Results and discussion

The anomaly detection results estimated by the D-MJPF at different abstraction levels, such as continuous and discrete levels, are discussed in this section. The SA functionality is embedded in the model's continuous level to detect the local abnormality. On the other hand, the

CA functionality to detect global abnormality is modeled into the discrete level.

7.3.1 Abnormality detection by D-MJPF

The area inside red dotted box indicates the 7.7, Fig. 7.8 and Fig. 7.9.

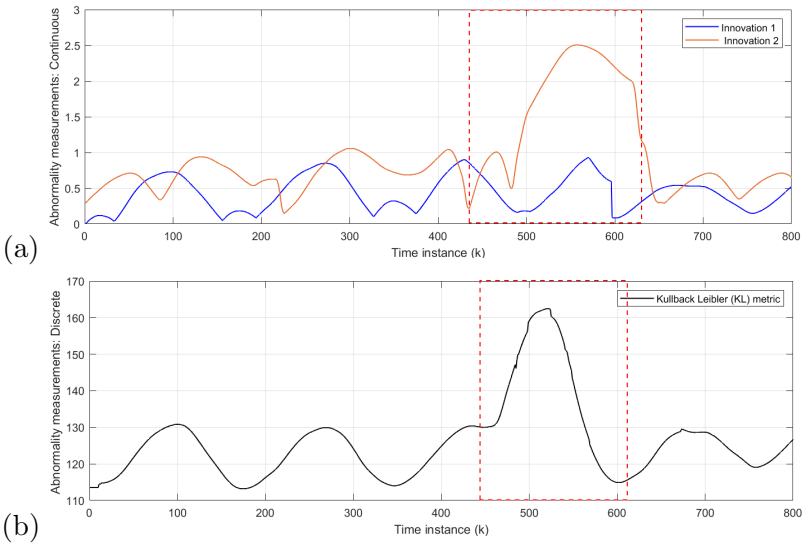


Figure 7.7: Abnormality measurements: Task 2 test data (a)Continuous level anomaly: The blue and orange plots belong to the anomaly estimated for *iCab1* and *iCab2*, respectively. The highest peaks of the orange plot represent the situation where the *iCab2* performed an emergency stop operation when it detected a pedestrian's presence. (b) Discrete level abnormality: The peaks show the global anomaly detected by the model.

Local abnormality detection: Self-awareness

The learned model is firstly tested with the rotor velocity datasets collected from Task 2 (refer to Fig. 7.5) of vehicles joint navigation. The continuous level part of the D-MJPF shown in Fig. 7.2 is dedicated to SA of the ego-things to predict future states and detects a local abnormality, i.e., the anomaly that happens around an individual ego-thing. The anomaly estimated by the innovation metric of the filter when tested with different datasets are plotted in Fig. 7.7 (a), Fig. 7.8 (a) and Fig. 7.9 (a).

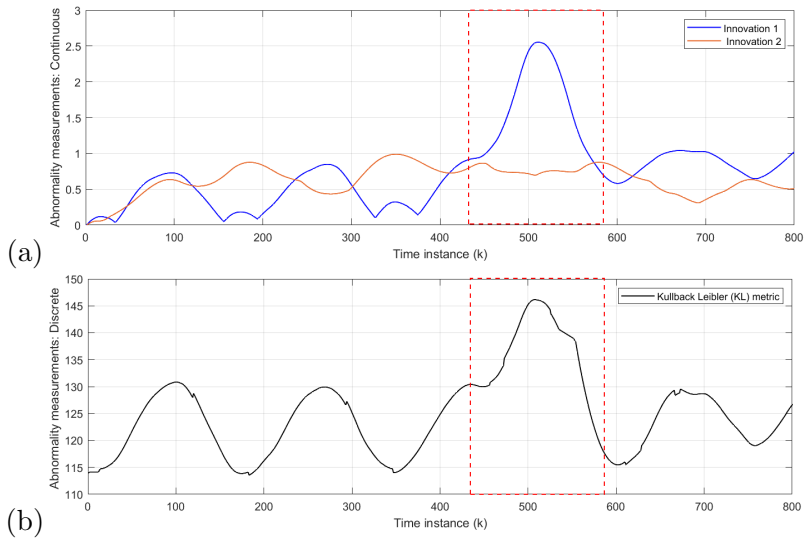


Figure 7.8: Abnormality measurements: Task 3 test data (a)Continuous level anomaly: The blue and orange plots belong to the anomaly estimated for *iCab1* and *iCab2*, respectively. The blue plot's highest peaks represent the situation where the *icab1* performed an emergency stop operation when it detected the presence of a pedestrian. (b) Discrete level abnormality: The peaks show the global anomaly detected by the model.

In Fig. 7.7 (a), plotted the result obtained by testing the initial model $M0$ (learned from Task 1 in Fig. 7.5) with the dataset of Task 2 in Fig. 7.5. When both of the iCab vehicles perform perimeter monitoring task, a random pedestrian appears in front of the *iCab2* vehicle, and it performs the emergency stop operation. However, iCab1 continues the task. Therefore the local anomaly detected by the model is only for the *iCab2* vehicle as shown in Fig. 7.7. The highest peak of *orange* plot inside the marked region represents the abnormal situation. As explained before, whenever the model detects an anomaly, a new model ($M1$) will learn incrementally to represent the new experience.

In the next step, we have tested the model $M1$ with another dataset collected from cooperative driving task 3 (Emergency stop 2) in Fig. 7.5. The anomaly results are shown in Fig. 7.8 (a). The blue and orange plot indicates the anomaly for *iCab1* and *iCab2*, respectively. In Task3 cooperative driving scenario, the pedestrian appears in front of the leader (*iCab1*) vehicle, and both vehicles perform emergency stop operation. Although both vehicles stopped with a pedestrian's

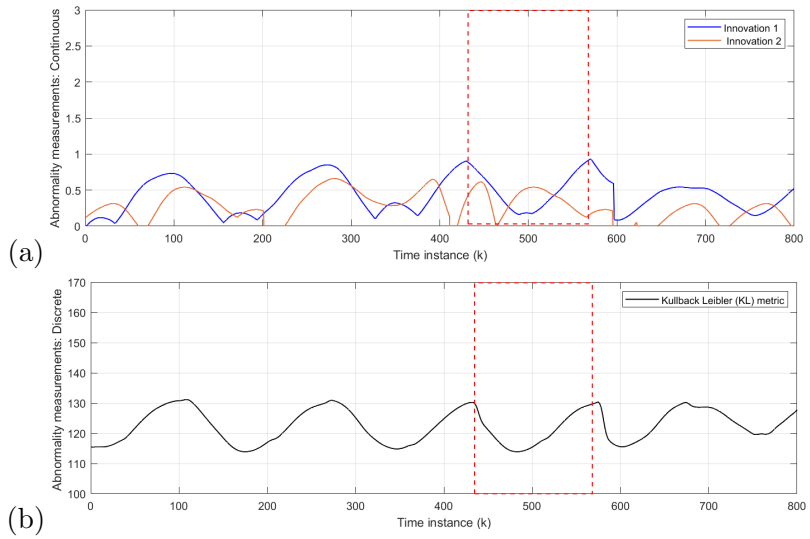


Figure 7.9: Abnormality measurements: Task 4 test data (a) The blue and orange plots belong to the anomaly estimated for *iCab1* and *iCab2*, respectively.(b) Discrete level abnormality: The peaks show the global anomaly detected by the model.

presence, only the anomaly is detected for *iCab1*. This is since *iCab2* already experienced this situation previously, and the existing part of the model was able to represent the current situation. This time, the incremental learning update was only for the *iCab1* and learned a new model called $M2$.

As a final test phase, the model $M2$ has been tested with another cooperative scenario dataset of Task 4 in Fig. 7.5. The anomaly results are plotted in Fig. 7.9. As expected, the model could represent the situation even if the pedestrian appeared in two spatial locations of the joint navigation task's trajectory. Because the existing model has experienced a similar situation in the past, there weren't any higher peaks in the anomaly signals. The model is learned from proprioceptive control data of rotor velocity (V) so that the models' performance is independent of spatial locations.

Global abnormality detection: Collective awareness

The global anomaly detection part belongs to the inference made by the discrete part of D-MJPF (orange shaded area in Fig. 7.2). The

model is able to detect the global anomaly that happens anywhere in the network. The global anomaly has been plotted in Fig. 7.7 (b), Fig. 7.8 (b) and Fig. 7.9 (b).

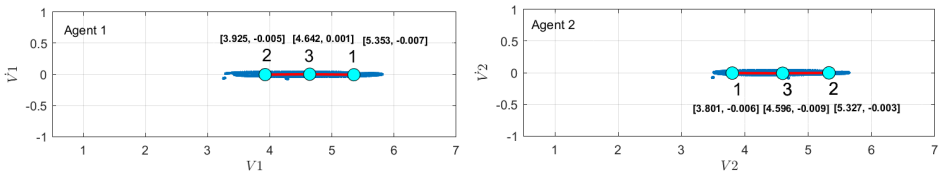
At first, the initial model ($M0$) was tested with the dataset collected from Task 2 shown in Fig. 7.5. The peaks in Fig. 7.7 (b) indicates the presence of pedestrian appeared in front of the *iCab2* vehicle when tested with the velocity dataset of Task 2 (refer Fig. 7.5). The discrete level part of the model is updated after the detection of the anomaly, and the new learned model is called $M1$. In the next step, the model ($M1$) tested with another dataset from Task 3 in Fig. 7.5. This time the pedestrian appears in front of the leader vehicle, and an emergency stop operation is performed by both vehicles (*iCab1* and *iCab2*). Therefore the interaction between the vehicles is slightly different than the one learned in the previous step. As a result, the model detected an anomaly and is plotted in Fig. 7.8 (b), and the model will be updated to the next version and is called $M2$.

Finally, the model $M2$ tested with the data of Task 4 (refer Fig. 7.5), and the resulting anomaly plotted in Fig. 7.9 (b). This time, there aren't any higher peaks as the model could represent this situation with the existing knowledge learned previously.

7.3.2 Interpretability and discussion

The interpretability module makes the reasoning behind the decision made by the model. Mainly we consider the model-based interpretability, i.e., the development of models that readily give insight into the relationships they have learned [70]. In this work, interpretability is defined as the capability of the model to detect the anomaly and use this information to correct the model incrementally.

The model's interpretability is explained with the intermediate level results such as clustering of GEs of anomaly data, vocabulary, and transition probability matrices. The individually clustered training data GEs to learn the initial model is shown in Fig. 7.10. The *cyan* colored nodes indicate the centroids of each cluster generated by the GNG algorithm. Each node is a 2D vector, and the values are marked in the plot.



(a) iCab1 (b) iCab2

Figure 7.10: Training data: Clustering of GEs.

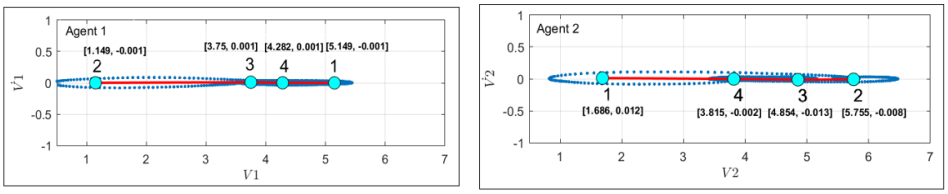
Continuous level

The model's continuous level is more focused on SA to detect the local anomalies. By testing the model with different scenario datasets, the system extracts knowledge when the model detects abnormalities and learn new models incrementally by exploiting the anomaly data. As stated before, the continuous level of the model is concerned about the individual ego-things experience. The limitation of the continuous level is, it doesn't care about the experience of other ego-things in the network. The discrete level that emphasizes collective awareness and global anomaly detection will help overcome the continuous level's limitations.

In the MJPF, each dynamic model in the continuous state is associated with one of the vocabulary variables at the discrete level. During the anomaly interval (when a pedestrian crosses in front of the iCab vehicles), the continuous states predicted by the model (with the chosen discrete variable's influence) will be different from the ground truth observed data. Such a difference is called GEs or abnormality and is used to learn new models incrementally. The generative DBN model's interpretability can be explained with the GEs (i.e., anomaly), discrete vocabulary, and transition probability matrices.

Once the model detects a local abnormality, the GEs data are stored (refer Fig. 7.3 and use this data to learn a new model incrementally by following the steps shown in Fig. 7.1. The main changes in the model w.r.t previous model will be in the transition probability (*green* arrows in Fig. 7.2) between the discrete level vocabulary variables.

The example of interpretability feature focuses on incrementally learning new models from the GEs of anomaly signal has been explained



(a) iCab1 (b) iCab2

Figure 7.11: Test data: Clustering of GEs. Node 2 belongs to iCab1and node 1 belongs to iCab2, represent the abnormality space.

below:

1. When the initial model $M0$ tested with the data collected from Task 2 (refer Fig. 7.5), the detected abnormality at the continuous level of the model is plotted in Fig. 7.7. The functionality of incremental learning of the new model will be enabled in this situation starting from the GEs/ abnormality signal. The model's continuous level detected abnormality only for *iCab2*. Therefore the model correction is required only for the part related to *iCab2*. The detected anomaly signal or GEs for *iCab2* is clustered and is plotted in Fig. 7.11b. With respect to the clusters produced of training data GEs as shown in Fig. 7.10b which was used to learn the initial model ($M0$), there is additional space activated GEs belongs to node 1 in Fig. 7.11b when clustered the detected anomaly.

   The discrete state-space belongs to node 1 in Fig. 7.11b encodes the information related to emergency stop operation when the pedestrian appeared. The GEs clusters' information will be used to learn discrete vocabulary and consequently update the transition probability matrix to learn the new model $M1$ incrementally.

2. Next, the model $M1$ will be tested with the data collected from Task 3 shown in Fig. 7.5. This time, the continuous abstraction level of the GDBN model detected anomaly for *iCab1* and is plotted in Fig. 7.8. Although both vehicles perform emergency stop operations, the anomaly was detected only for the leader vehicle (iCab) because the follower (iCab2) was already experienced a similar situation previously. Therefore the model

was well able to represent the behaviors of iCab2. This time the model correction only required for the part related to iCab1. The GEs produced from the anomaly data for iCab1 is plotted in Fig. 7.11a. With respect to the GEs cluster for iCab1 in the training phase as shown in Fig. 7.10a, the GEs clustering produced from the Task 3 test data enables additional clustering space belongs to node 2 in Fig. 7.11a. Then the parameters extracted from the clusters are used to generate discrete vocabulary and transition probability matrix to learn a new model called $M2$.

3. Finally, the model $M2$ tested with the data collected from task 4 (emergency stop 3) in Fig. 7.5. This time, a random pedestrian appeared two times in the entire trajectory of the vehicles' maneuvering operation. However, the model has not detected anomaly at continuous level (refer Fig. 7.9 (a)) because the existing model was well able to represent the current situation and well predicts the state of the vehicles.

Whenever the ego-things experience a new situation that was not seen in the past, the system extracts knowledge and updates/learns a new model to represent the new situation. So that if we consider a particular stage of the filter, it will be able to describe all the previous experiences passed by the ego-things. If similar situations occur in the future, the models would easily represent them and help to make appropriate decisions.

Discrete level

Contrary to the continuous level (sub-symbolic level) of the model that can detect a local anomaly, the model's discrete level (symbolic level) will mainly focus on the global anomaly detection, i.e., the abnormality that happens around any ego-thing in the network. The continual learning of new models when the system passes through new joint experiences is part of the interpretability feature. With this CA functionality, the model inside one ego-thing can detect other ego-thing anomalies in the network. It would help the model to make appropriate decisions in different situations.

The interpretability can be visible by checking the intermediate results. The process of continual learning of new model from the GEs of anomaly data is similar as explained in section 7.3.2. Once the model detected the global anomaly at the model's discrete abstraction level, the GEs produced from the local anomaly signal will be clustered and use this information first to update the individual transition probability matrices. Then the changes will be updated to the next level of joint vocabulary (words) and joint transition probability matrix (represented by green arrows in Fig. 7.2). Therefore, if the system faces similar situations in the future, the model's discrete abstraction level would be well able to represent the interaction between the ego-things.

This chapter has used four different scenario datasets for model learning and testing the model's fitness. The proposed method has many advantages as well as limitations.

Capabilities:

- The proposed method can be used in a large network consists of more ego-things (< two ego-things).
- In this work, we have mainly considered one data variable, such as rotor velocity. However, the method can be applied for other low dimensional exteroceptive and proprioceptive sensory data variables and combinations of data variables to develop models representing co-operative tasks of ego-things.

Limitations:

- The proposed method is designed to develop models from low-dimensional sensory data. To use high dimensional data, need modifications in the method, and also it may be required to use deep learning algorithms.
- The interpretability should be improved by presenting the results obtained from different abstraction levels of the model. Furthermore, the interpretability can be enriched by comparing the differences and similarities of the models that are learned continually when the ego-things pass through new experiences.

– The ego-things in this work are autonomous vehicles. Therefore, to use in other ego-things such as drones and other IoT devices required modifications in the proposed method.

Future research directions could incorporate multi-sensory data to develop multi-modal interpretable ML models. The interpretability could be achieved by exploring graph matching techniques which can be used to compare the models learned incrementally while ego-things pass through different experiences. The graph matching techniques [26, 107, 65] can be exploited to compare the intermediate results (such as discrete vocabulary and transition probability matrices) obtained in different models and make the reasoning behind the model detected abnormal situations.

Another important research direction could be considering different communication protocols to exchange data/parameters among ego-things and compare the performances by giving importance to energy efficiency. Moreover, automatic classification of abnormalities can be considered in the future to improve the model performance.

7.4 Chapter summary

This chapter presented a method to develop an interpretable machine learning model for the agents' network. The data-driven model has the SA and CA functionalities to detect local and global anomalies. A D-MJPF is used to make inferences on the generative DBN model at different abstraction levels. The interpretability is closely related to the continual incremental learning of the models when an abnormal situation occurs. The intermediate results which are related to the interpretability are presented as discussed. The interpretability part is given importance in describing how the models make abnormality detection decisions and how the anomaly data has been exploited to learn new models incrementally. The model has been tested with different scenario datasets, and the obtained abnormality detection results at different abstraction levels are presented and discussed. Finally, a discussion about the capabilities and limitations of the proposed method has been conducted.

Chapter 8

Conclusions and Future work

This chapter summarizes and evaluates the results of the previous chapters' concerning the research objectives set out before. Moreover, the chapter concludes by discussing the possible future research directions to the accurate joint prediction of unseen dynamics of a network of agents.

8.1 Summary of main achievements

With respect to the research questions presented in Chapter 1, the thesis identifies the below contributions:

1. Chapter 4 of this thesis proposes a methodology to develop agents' self-awareness by giving particular attention to detect abnormal situations. The proposed method focuses on selecting the most precise DBN model when predicting abnormalities in real scenarios where multiple sensory data is analyzed. The learned model is data-driven, and the method is evaluated with real experimental data taken from a moving vehicle performing some tasks in a closed environment. Results suggest that the proposed method recognizes the features from a set of sensory data that facilitates the recognition of previously unseen maneuvers, i.e., abnormal

situations. The proposed method can be useful for selecting relevant features when dealing with a large number of features in networking operations.

2. Chapter 5 of this thesis proposed several methods to recognize abnormal situations in an agent network. Each agent learns a set of DBN models describing the normal behavior of self and all the other entities in the network. A MJPF is employed to infer the future states of the entities. The abnormality measurement values calculated in each of the DBN models suggest that the proposed method provides good performance in detecting environmental abnormalities. Moreover, information exchange among entities has been considered to enhance the proposed strategy.

   The considered test scenario comprises two smart vehicles, one following the other, which move along a predefined track. Communication performance has been collected to verify the reliability of the data exchange, quantify the expected performance in terms of delay and loss and consider how these performances could affect the abnormality detection process. The DBN model performance is investigated when each object communicates the ground truth observations to the other entities in the network. To compare the performance with different parameters of the considered channel model (Rician model), such as K-factor, distance, and data rates, ROC curves were plotted, and the reliability (AUC) and ACC metrics calculated. The AUC and ACC values decreased when the data rates increased. Moreover, when the environment changed from the ideal case (i.e., no loss case) to an urban scenario, the model's performance degraded further.

3. Chapter 6 of this thesis presented a method to develop multimodal collective awareness for networked IoT nodes performing collaborative tasks. The IoT nodes in this work are autonomous vehicles, and each of the vehicles is assumed to be having machine learning capabilities. The CDBN models learned from exteroceptive and proprioceptive sensory data have the functionality to extract unique features of the system related to self and collective awareness and detect abnormalities happening anywhere in the networked ego-things. The CDBN models are data-driven

and capable of predicting distributed states and detecting abnormalities at different abstraction levels. The distributed state estimation is performed by D-MJPF associated with each CDBN model.

The models inside each agent can synchronously estimate the possible abnormalities around any of the agents in the network. Moreover, the models can describe abnormality related to single specific components of the vector used for model learning; this is an additional explainability feature of the models. In the offline training phase, the multi-sensory data collected when the agents perform a joint task is used to learn the CDBN models. In the online test phase, the model's fitness tested with the data sets from a new collaborative task different than the one used in the training phase. The presented results at different abstraction levels provide evidence for the proposed method's efficiency in detecting abnormal situations in the networked agents. Moreover, the effects of wireless communication channels on the model performance were analyzed by considering different communication protocols and channel conditions. Finally, the obtained abnormality results were compared by different performance evaluation metrics (MSE, ACC, and F1 score).

4. Chapter 7 presents a method to develop an interpretable machine learning model for the agents' network. The data-driven model has the self-awareness and collective awareness functionalities to detect local and global anomalies. A D-MJPF is used to make inferences at different abstraction levels of the model.

   The interpretability is closely related to the continual incremental learning of the models when an abnormal situation occurs. The local and global abnormality detection results are presented and compared.

8.2 Future work

This section provides the potential future research directions that could be conducted for the methods described in the previous chapters

of this thesis.

1. In this thesis, mainly three different metrics to estimate abnormality were investigated using MJPF, such as Innovation, Hellinger distance, and Kullback Leibler divergence. In the future, it is important to embed other distance metrics to estimate abnormality and compare the abnormality results.
2. The collective awareness data-driven models can include more functionalities to extract more networked ego-things features by including different cooperative task scenarios and more data variables and combinations of variables. Moreover, this work mainly considered low dimensional data to develop models, and it is important to check what are the changes required in the model to use it for high dimensional data such as images, lidar, etc
3. Another future work could be developing collective awareness interaction models for an ego-things network by giving importance to anomaly detection at different abstraction levels.
4. Autonomous decision-making based on the different abnormalities detected by the model is important for future work. Also, classification of abnormality is an important task to be embedded in the model. The abnormality detection performed by the models built from the agents' exteroceptive and proprioceptive sensory data would focus on the different features and could be classified into different categories. For instance, the exteroceptive sensory data focuses on contextual anomaly. On the other hand, the proprioceptive sensory data gives importance to the agent's internal features, and anomaly happens due to internal changes. Therefore, classifying the anomaly based on the system's different features would help improve the performance while the agents face an internal or external abnormal situation.
5. Add functionalities to provide energy efficiency using different communication protocols and parameter settings of the simulation model. The development of an energy-aware module is important to evaluate the system's energy efficiency based on different parameter settings, communication protocols, etc. Including

such a module can improve the system's reliability, and it would be easier to choose the model for different application areas.

6. The model performance has been evaluated using different communication protocols implemented in a simulated environment in this thesis. In the future, this could be performed in a real-time environment to improve the model's reliability.

Bibliography

[1] Ieee standard for information technology– local and metropolitan area networks– specific requirements– part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications amendment 6: Wireless access in vehicular environments. *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009)*, pages 1–51, July 2010.

[2] Hervé Abdi and Neil J Salkind. Encyclopedia of measurement and statistics. *Thousand Oaks, CA: Sage. Agresti, A.(1990) Categorical data analysis., New York: Wiley. Agresti, A.(1992) A survey of exact inference for contingency tables. Statist Sci*, 7:131–153, 2007.

[3] S. M. Shafiul Alam, Balasubramaniam Natarajan, and Anil Pahwa. Distributed agent-based dynamic state estimation over a lossy network. In Giancarlo Fortino, Stamatis Karnouskos, and Pedro José Marrón, editors, *Proceedings of the 5th International Workshop on Networks of Cooperating Objects for Smart Cities (UBICITEC 2014) co-located with CPSWeek 2014, Berlin, Germany, Apr 14, 2014*, volume 1156 of *CEUR Workshop Proceedings*, pages 1–15. CEUR-WS.org, 2014.

[4] SM Shafiul Alam, Balasubramaniam Natarajan, and Anil Pahwa. Distributed agent-based dynamic state estimation over a lossy network. In *UBICITEC*, pages 1–15, 2014.

[5] SM Shafiul Alam, Balasubramaniam Natarajan, and Anil Pahwa. Agent based optimally weighted kalman consensus fil-

ter over a lossy network. In *2015 IEEE global communications conference (GLOBECOM)*, pages 1–6. IEEE, 2015.

[6] Brian DO Anderson and John B Moore. *Optimal filtering*. Courier Corporation, 2012.

[7] Jens B Asendorpf, Veronique Warkentin, and Pierre-Marie Baudonniere. Self-awareness and other-awareness. ii: Mirror self-recognition, social contingency awareness, and synchronic imitation. *Developmental Psychology*, 32(2):313, 1996.

[8] Jane Bajgar, Joseph Ciarrochi, Richard Lane, and Frank P Deane. Development of the levels of emotional awareness scale for children (leas-c). *British Journal of Developmental Psychology*, 23(4):569–586, 2005.

[9] Smith Baker. The identification of the self. *Psychological Review*, 4(3):272, 1897.

[10] Bhashyam Balaji and Karl Friston. Bayesian state estimation using generalized coordinates. In *Signal Processing, Sensor Fusion, and Target Recognition XX*, volume 8050, page 80501Y. International Society for Optics and Photonics, 2011.

[11] Yaakov Bar-Shalom, Thomas E Fortmann, and Peter G Cable. Tracking and data association. *Acoustical Society of America Journal*, 87(2):918–919, 1990.

[12] Yaakov Bar-Shalom and Xiao-Rong Li. *Multitarget-multisensor tracking: principles and techniques*, volume 19. YBs Storrs, CT, 1995.

[13] Allen L Barker, Donald E Brown, and Worthy N Martin. Bayesian estimation and the kalman filter. *Computers & Mathematics with Applications*, 30(10):55–77, 1995.

[14] M Baydoun, D Campo, V Sanguineti, L Marcenaro, A Cavallaro, and C Regazzoni. Learning switching models for abnormality detection for autonomous driving. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 2606–2613. IEEE, 2018.

[15] M. Baydoun, M. Ravanbakhsh, D. Campo, P. Marin, D. Martin, L. Marcenaro, A. Cavallaro, and C. S. Regazzoni. a multi-

perspective approach to anomaly detection for self -aware embodied agents. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6598–6602, 2018.

[16] Mohamad Baydoun, Damian Campo, Divya Kanapram, Lucio Marcenaro, and Carlo S Regazzoni. Prediction of multi-target dynamics using discrete descriptors: an interactive approach. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3342–3346. IEEE, 2019.

[17] Mohamad Baydoun, Mahdyar Ravanbakhsh, Damian Campo, Pablo Marin, David Martin, Lucio Marcenaro, Andrea Cavallaro, and Carlo S Regazzoni. A multi-perspective approach to anomaly detection for self-aware embodied agents. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6598–6602. IEEE, 2018.

[18] Alessandro Bazzi, Barbara M. Masini, and Alberto Zanella. Cooperative awareness in the internet of vehicles for safety enhancement. *EAI Endorsed Transactions on Internet of Things*, 3(9), 8 2017.

[19] Rudolf Beran et al. Minimum hellinger distance estimates for parametric models. *The annals of Statistics*, 5(3):445–463, 1977.

[20] Christian Berger. From a competition for self-driving miniature cars to a standardized experimental platform: concept, models, architecture, and evaluation. *arXiv preprint arXiv:1406.7768*, 2014.

[21] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943.

[22] Richard Bishop. Intelligent vehicle applications worldwide. *IEEE Intelligent Systems and Their Applications*, 15(1):78–81, 2000.

[23] H. A. P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with markovian switching coeffi-

cients. *IEEE Transactions on Automatic Control*, 33(8):780–783, 1988.

[24] A. Bourazeri and J. Pitt. Collective awareness for collective action in socio-technical systems. In *2014 IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems Workshops*, pages 90–95, 2014.

[25] Aikaterini Bourazeri and Jeremy Pitt. Collective awareness for collective action in socio-technical systems. *Proceedings - 2014 IEEE 8th International Conference on Self-Adaptive and Self-Organizing Systems Workshops, SASOW 2014*, pages 90–95, 03 2015.

[26] Tibério S Caetano, Julian J McAuley, Li Cheng, Quoc V Le, and Alex J Smola. Learning graph matching. *IEEE transactions on pattern analysis and machine intelligence*, 31(6):1048–1058, 2009.

[27] Damian Campo, Alejandro Betancourt, Lucio Marcenaro, and Carlo Regazzoni. Static force field representation of environments based on agents' nonlinear motions. *EURASIP Journal on Advances in Signal Processing*, 2017(1):13, 2017.

[28] Xiangtuo Chen and Paul-Henry Cournéde. Model-driven and data-driven approaches for crop yield prediction: analysis and comparison. *International Journal of Mathematical and Computational Sciences*, 11(7):334–342, 2018.

[29] Ketan Devadiga. Ieee 802.15. 4 and the internet of things. *Aalto University School of Science*, 2007.

[30] Guoru Ding, Qihui Wu, Linyuan Zhang, Yun Lin, Theodoros A Tsiftsis, and Yu-Dong Yao. An amateur drone surveillance system based on the cognitive internet of things. *IEEE Communications Magazine*, 56(1):29–35, 2018.

[31] A. Doucet, N. J. Gordon, and V. Krishnamurthy. Particle filters for state estimation of jump markov linear systems. *IEEE Transactions on Signal Processing*, 49(3):613–624, 2001.

[32] Arnaud Doucet, Neil J Gordon, and Vikram Krishnamurthy. Particle filters for state estimation of jump markov linear sys-

tems. *IEEE Transactions on signal processing*, 49(3):613–624, 2001.

[33] H. Driessen and Y. Boers. Efficient particle filter for jump markov nonlinear systems. *IEE Proceedings - Radar, Sonar and Navigation*, 152(5):323–326, 2005.

[34] Shelley. Duval and Robert A Wicklund. *A theory of objective self awareness [by] Shelley Duval and Robert A. Wicklund*. Academic Press New York, 1972.

[35] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[36] V Fox, Jeffrey Hightower, Lin Liao, Dirk Schulz, and Gaetano Borriello. Bayesian filtering for location estimation. *IEEE pervasive computing*, 2(3):24–33, 2003.

[37] Karl Friston and Stefan Kiebel. Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 364(1521):1211–1221, 2009.

[38] Bernd Fritzke. A self-organizing network that can follow non-stationary distributions. In *International conference on artificial neural networks*, pages 613–618. Springer, 1997.

[39] Bernd Fritzke et al. A growing neural gas network learns topologies. *Advances in neural information processing systems*, 7:625–632, 1995.

[40] Mario Gerla, Eun-Kyu Lee, Giovanni Pau, and Uichin Lee. Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds. In *2014 IEEE world forum on internet of things (WF-IoT)*, pages 241–246. IEEE, 2014.

[41] Andrea Goldsmith. *Wireless communications*. Cambridge university press, 2005.

[42] D Gomez, P Marin-Plaza, Ahmed Hussein, A Escalera, and JM Armingol. Ros-based architecture for autonomous intelligent campus automobile (icab). *UNED Plasencia Revista de Investigacion Universitaria*, 12:257–272, 2016.

[43] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F - Radar and Signal Processing*, 140(2):107–113, April 1993.

[44] Caroline Goukens, Siegfried Dewitte, and Luk Warlop. Me, myself, and my choices: The influence of private self-awareness on choice. *Journal of Marketing Research*, 46(5):682–692, 2009.

[45] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[46] Juan Antonio Guerrero-Ibanez, Sherali Zeadally, and Juan Contreras-Castillo. Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and internet of things technologies. *IEEE Wireless Communications*, 22(6):122–128, 2015.

[47] Vijay Gupta, Demetri Spanos, Babak Hassibi, and Richard M Murray. On lqg control across a stochastic packet-dropping link. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 360–365. IEEE, 2005.

[48] C. Han, G. Feng, and H. Zhang. Optimal markov jump filter for stochastic systems with markovian transmission delays. In *Proceedings of the 30th Chinese Control Conference*, pages 4566–4571, 2011.

[49] Seul-Ki Han, Won-Sang Ra, and Jin Bae Park. Tdoa/fdoa based target tracking with imperfect position and velocity data of distributed moving sensors. *International Journal of Control, Automation and Systems*, 15(3):1155–1166, 2017.

[50] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[51] D. Kanapram, D. Campo, M. Baydoun, L. Marcenaro, E. L. Bodanese, C. Regazzoni, and M. Marchese. Dynamic bayesian approach for decision-making in ego-things. In *2019 IEEE 5th*

World Forum on Internet of Things (WF-IoT), pages 909–914, April 2019.

[52] D. T. Kanapram, M. Marchese, E. L. Bodanese, D. M. Gomez, L. Marcenaro, and C. Regazzoni. Dynamic bayesian collective awareness models for a network of ego-things. *IEEE Internet of Things Journal*, pages 1–1, 2020.

[53] D. T. Kanapram, F. Patrone, P. Marin-Plaza, M. Marchese, E. L. Bodanese, L. Marcenaro, D. M. Gomez, and C. Regazzoni. Collective awareness for abnormality detection in connected autonomous vehicles. *IEEE Internet of Things Journal*, pages 1–1, 2020.

[54] Divya Kanapram, Pablo Marin-Plaza, Lucio Marcenaro, David Martin, Arturo de la Escalera, and Carlo Regazzoni. Self-awareness in intelligent vehicles: Experience based abnormality detection. In *Iberian Robotics conference*, pages 216–228. Springer, 2019.

[55] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. The one simulator for dtn protocol evaluation. In *Proceedings of the 2nd international conference on simulation tools and techniques*, page 55. ICST (Institute for Computer Sciences, Social-Informatics and . . . , 2009.

[56] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[57] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[58] Daphne Koller and Uri Lerner. Sampling in factored dynamic systems. In *Sequential Monte Carlo Methods in Practice*, 2001.

[59] Jitendra Kumar, Vishal Goyal, and Devbrat Gupta. Study of network-induced delays on networked control systems. In *Advances in Data and Information Sciences*, pages 13–21. Springer, 2020.

[60] Erich L Lehmann and George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

[61] Wenling Li, Yingmin Jia, and Junping Du. Distributed kalman consensus filter with intermittent observations. *Journal of the Franklin Institute*, 352(9):3764–3781, 2015.

[62] Rodolfo Lourenzutti and Renato A Krohling. The hellinger distance in multicriteria decision making: An illustration to the topsis and todim methods. *Expert Systems with Applications*, 41(9):4414–4421, 2014.

[63] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark. Connected vehicles: Solutions and challenges. *IEEE Internet of Things Journal*, 1(4):289–299, Aug 2014.

[64] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[65] Hermina Petric Maretic, Mireille EL Gheche, Giovanni Chierchia, and Pascal Frossard. Got: An optimal transport framework for graph comparison. *arXiv preprint arXiv:1906.02085*, 2019.

[66] Pablo Marin-Plaza, Ahmed Hussein, David Martin, and Arturo de la Escalera. Global and local path planning study in a ros-based research platform for autonomous vehicles. *Journal of Advanced Transportation*, 2018, 2018.

[67] Thomas Martinetz and K. Schulten. A "neural-gas" network learns topologies. *Artificial neural networks*, 1:397–402, 01 1991.

[68] Yilin Mo and Bruno Sinopoli. Kalman filtering with intermittent observations: Tail distribution and critical value. *IEEE Transactions on Automatic Control*, 57(3):677–689, 2011.

[69] Alain Morin. Levels of consciousness and self-awareness: A comparison and integration of various neurocognitive views. *Consciousness and cognition*, 15(2):358–371, 2006.

[70] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, 2019.

[71] Kevin Murphy, Saira Mian, et al. Modelling gene expression data using dynamic bayesian networks. Technical report, Technical report, Computer Science Division, University of California ..., 1999.

[72] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[73] Dénes Némedi. Collective consciousness, morphology, and collective representations: Durkheim's sociology of knowledge, 1894-1900. *Sociological Perspectives*, 38(1):41–56, 1995.

[74] Reza Olfati-Saber. Kalman-consensus filter: Optimality, stability, and performance. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 7036–7042. IEEE, 2009.

[75] H. A. P. Blom. An efficient filter for abruptly changing systems. In *The 23rd IEEE Conference on Decision and Control*, pages 656–658, 1984.

[76] Bryan Parno and Adrian Perrig. Challenges in securing vehicular networks. In *Workshop on hot topics in networks (HotNets-IV)*, pages 1–6. Maryland, USA, 2005.

[77] Vladimir Ivan Pavlovic and Thomas S Huang. *Dynamic bayesian networks for information fusion with applications to human-computer interfaces*, volume 44. University of Illinois at Urbana-Champaign, 1999.

[78] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[79] Kurt Plarre and Francesco Bullo. On kalman filtering for detectable systems with intermittent observations. *IEEE Transactions on Automatic Control*, 54(2):386–390, 2009.

[80] D. M. W. Powers. Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011.

[81] Schriftliche Prüfungsarbeit and Guillermo S. Donatti. Finding optimal parameters for neural gas networks using evolutionary algorithms. 2009.

[82] M. Ravanbakhsh, M. Baydoun, D. Campo, P. Marin, D. Martin, L. Marcenaro, and C. S. Regazzoni. Hierarchy of gans for learning embodied self-awareness model. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1987–1991, 2018.

[83] M. Ravanbakhsh, M. Baydoun, D. Campo, P. Marin, D. Martin, L. Marcenaro, and C. S. Regazzoni. Learning multi-modal self-awareness models for autonomous vehicles from human driving. In *2018 21st International Conference on Information Fusion (FUSION)*, pages 1866–1873, 2018.

[84] C. S. Regazzoni, L. Marcenaro, D. Campo, and B. Rinner. Multisensorial generative and descriptive self-awareness models for autonomous systems. *Proceedings of the IEEE*, 108(7):987–1010, 2020.

[85] B. Rinner, L. Esterle, J. Simonjan, G. Nebehay, R. Pflugfelder, G. Fernández Domínguez, and P. R. Lewis. Self-aware and self-expressive camera networks. *Computer*, 48(7):21–28, 2015.

[86] DB Rubin. A noniterative sampling/importance resampling alternative to data augmentation for creating a few imputations when fractions of missing information are modest: The sir algorithm. *Journal of the American Statistical Association*, 82:544–546, 1987.

[87] Luca Schenato, Bruno Sinopoli, Massimo Franceschetti, Kameshwar Poolla, and S Shankar Sastry. Foundations of control and estimation over lossy networks. *Proceedings of the IEEE*, 95(1):163–187, 2007.

[88] Mark Senn. *HMMs and Particle Filtering*, 2012 (accessed September 9, 2020). `https://inst.eecs.berkeley.edu/~cs188/sp12/slides/cs188%20lecture%2017%20--%20HMMs%20and%20particle%20filters%202PP.pdf`.

[89] Omer Berat Sezer, Erdogan Dogdu, and Ahmet Murat Ozbayoglu. Context-aware computing, learning, and big data in internet of things: a survey. *IEEE Internet of Things Journal*, 5(1):1–27, 2017.

[90] Dan Simon. Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. *IET Control Theory & Applications*, 4(8):1303–1318, 2010.

[91] Amit Singhal and Christopher R Brown. Dynamic bayes net approach to multimodal sensor fusion. In *Sensor Fusion and Decentralized Control in Autonomous Robotic Systems*, volume 3209, pages 2–11. International Society for Optics and Photonics, 1997.

[92] Bruno Sinopoli, Luca Schenato, Massimo Franceschetti, Kameshwar Poolla, Michael I Jordan, and Shankar S Sastry. Kalman filtering with intermittent observations. *IEEE transactions on Automatic Control*, 49(9):1453–1464, 2004.

[93] G. Slavic, D. Campo, M. Baydoun, P. Marin, D. Martin, L. Marcenaro, and C. Regazzoni. Anomaly detection in video data based on probabilistic latent space models. In *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8, 2020.

[94] Enbin Song, Jie Xu, and Yunmin Zhu. Optimal distributed kalman filtering fusion with singular covariances of filtering errors and measurement noises. *IEEE Transactions on Automatic Control*, 59(5):1271–1282, 2014.

[95] GA Tawney. Feeling and self-awareness. *Psychological Review*, 9(6):570, 1902.

[96] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, Inc., Orlando, FL, USA, 4th edition, 2008.

[97] Anna Maria Vegni and Valeria Loscri. A survey on vehicular social networks. *IEEE Communications Surveys & Tutorials*, 17(4):2397–2419, 2015.

[98] Sergio Verdú. Total variation distance and the distribution of relative information. In *2014 Information Theory and Applications Workshop (ITA)*, pages 1–3. IEEE, 2014.

[99] Nannan Wang, Mingrui Zhu, Jie Li, Bin Song, and Zan Li. Data-driven vs. model-driven: Fast face sketch synthesis. *Neurocomputing*, 257:214–221, 2017.

[100] G Welch and G Bishop. An introduction to the kalman filter, university of north carolina at chapel hill, department of computer science, chapel hill, nc, usa. *TR95-041*, 1:995, 1995.

[101] Alan F. T. Winfield. *Robots with Internal Models: A Route to Self-Aware and Hence Safer Robots*, chapter Chapter 16, pages 237–252.

[102] Michael Wooldridge. *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition, 2009.

[103] Guotao Xie, Hongbo Gao, Bin Huang, Lijun Qian, and Jianqiang Wang. A driving behavior awareness model based on a dynamic bayesian network and distributed genetic algorithm. *International Journal of Computational Intelligence Systems*, 11(1):469–482, 2018.

[104] Guangming Xiong, Peiyun Zhou, Shengyan Zhou, Xijun Zhao, Haojie Zhang, Jianwei Gong, and Huiyan Chen. Autonomous driving of intelligent vehicle bit in 2009 future challenge of china. In *Intelligent Vehicles Symposium (IV), 2010*, pages 1049–1053. IEEE, 2010.

[105] Ning Yang, Wei Feng Tian, Zhi Hua Jin, and Chuan Bin Zhang. Particle filter for sensor fusion in a land vehicle navigation system. *Measurement science and technology*, 16(3):677, 2005.

[106] W. Youn and H. Myung. Robust interacting multiple model with modeling uncertainties for maneuvering target tracking. *IEEE Access*, 7:65427–65443, 2019.

[107] Hassan Zaal, Mohamad Baydoun, Lucio Marcenaro, Laurissa Tokarchuk, and Carlo S Regazzoni. Abnormality detection using graph matching for multi-task dynamics of autonomous systems. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8. IEEE, 2019.

[108] S. Zhu, T. S. Ghazaany, S. M. R. Jones, R. A. Abd-Alhameed, J. M. Noras, T. Van Buren, J. Wilson, T. Suggett, and S. Marker. Probability distribution of rician k-factor in urban, suburban and rural areas using real-world captured data. *IEEE Transactions on Antennas and Propagation*, 62(7):3835–3839, July 2014.