Adaptive and Tractable Bayesian Context Inference for Resource Constrained Devices

Dipl. Inform. Korbinian Frank

Primary Supervisor: **Dr. Tom Pfeifer**
Secondary Supervisor: **Dr. Patrick Robertson**

In partial fulfilment of the requirements for the award of
Doctor of Philosophy

Department of Computing, Mathematics and Physics

Waterford Institute of Technology
Ireland

October 2011

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work and has not been taken from the work of others save to the extent that such work has been cited and acknowledged within the text of my work.

Signed:.....
Date: June 2011

Adaptive and Tractable Bayesian Context Inference for Resource Constrained Devices

Dipl. Inform. Korbinian Frank

Primary Supervisor: Dr. Tom Pfeifer
Secondary Supervisor: Dr. Patrick Robertson

Abstract

Context inference is necessary in ubiquitous computing to provide information about contextual information which is not directly measurable from sensors or obtained from other information sources. Server based, central inference would not scale due to the expected amount of context requests. Mobile, distributed context inference faces problems because of the high computational complexity of inference mechanisms. Bayesian inference techniques are particularly well suited, as they allow for more flexible modelling of situations than propositional logic, are always decidable as opposed to higher order logics, are intelligible to humans as opposed to neural networks and allow for uncertain or missing information. As inference in them however is NP-hard, methods have to be introduced to fit them to the requirements of ubiquitous computing and mobile, resource constrained devices.

To this end, this work proposes to divide Bayesian networks for context inference into modules, called Bayeslets. Bayeslets can be composed among each other to fulfil an inference request via interface nodes about which additional assumptions are made: Considering input nodes as observed, more efficient inference methods can possibly be applied and by defining explicit output nodes for connection, a relevancy based dynamic composition of Bayeslets can be realised, so the evaluated number of Bayeslets always stays at a minimum. The inference time of Bayeslets can be further reduced by adapting edges and value ranges to the user's personal requirements and the current situation. The application of these concepts is shown in general examples of high level context used in the user's smart space, in his work environment, as well as in road traffic. Experimental results show that this process results in a significant reduction of the inference load. The Bayeslets for location and human motion related activity are of particular importance for context awareness and therefore considered and evaluated in detail.

The set of tools proposed in this thesis allows to apply a fully Bayesian approach to context inference, fulfilling the requirements of ubiquitous computing and mobile, resource constrained devices.

Acknowledgements

The work on this thesis has been carried out at the Institute of Communications and Navigation of the German Aerospace Center (DLR) in close cooperation with the Telecommunications Software & Systems Group (TSSG) of the WIT. Thereby my research has received funding from the European Community's Sixth and Seventh Framework Programme under the Collaborative Projects *Societies* (Self Orchestrating CommunIty ambiEnT IntelligEnce Spaces), Persist (PERsonal Self-Improving SmarT spaces) and Daidalos II (Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimised personal Services - II), as well as from the Irish HEA through the PRTLI cycle 4 project "Serving Society: Management of Future Communication Networks and Services".

I want to extend my gratitude to all former and current colleagues, in both institutions as well as in the projects, forming the work environment which did not only advance my research, but also allowed for a good working atmosphere. In particular I appreciate the help of Martin Hammer, Jutta Uelner and Christina Burger in all administrative issues, making my life at DLR much easier.

My current and former colleagues Bernhard Krach, Matthias Röckl, Michael Angermann and Mohammed Khider have been a constant help throughout our common work. Their expertise in Bayesian methods, their tool sets and implementations have been as much support for me as our discussions, the mixture of computer science and electrical engineering view points and their examples of how to successfully complete a PhD thesis. Uwe Fiebig has been a challenging and always supporting head of department and is credited for the scientific steering and the excellent research conditions of the Communication Systems department at DLR.

The students who performed their master's thesis work must be credited for their contributions to this thesis. I want to thank in particular Sergio Fortes Rodríguez and María José Vera Nadales for their exceptional degree of creativity during their work.

I am deeply grateful to my supervisors Tom Pfeifer and Patrick Robertson for their excellent advice and guidance throughout the work on my thesis. Their support in administrative issues, their instructions for scientific writing, their ideas for research and publication possibilities, and not least their belief in my work and in my ability to complete this work have been major factors to its success.

Furthermore, I want to thank Declan O'Sullivan and Sven van der Meer for their willingness to act as examiners for this research, the comforting atmosphere during the PhD viva voce examination and their advice how to further pursue this research.

Last, I want to express my heartiest gratitude and love to my family. My parents Ludwig and Angelika and my brothers Rupert and Martin provided the supporting family structures and the unconditional backing without which my personal and professional formation would not have been possible. My fiancé Cristina has never lost faith in me, gave me hope and inspiration. Her support at home and at work was invaluable for the success of this thesis. I hope I will be able to return the same value for all her upcoming tasks.

Contents

Chapter 1

Introduction and Motivation

Twelve years ago in 1999, right after school in Germany, I acquired my first mobile phone. And I am a techie and have always been so at school. Always longing for the newest technical innovation. Only a couple of years later at university, you had to justify yourself for not being able to present a mobile phone number at which one can always reach you. The first years at work and you noticed the sensation that being always reachable by phone can be a burden. Anyway, the boom of mobile communications went on. First location aware services were being used, Java applications came up. Network bandwidth is getting cheaper, along with MEMS based sensors and GPS receivers. In 2007 Apple's iPhone prepares the ground for the smart phone generation with data flat rates, many internal sensors and almost unlimited access to information from the internet. For all mobile operating systems, application stores sprout and grow day by day. Technically, we are not so far away from the vision of pervasive, context aware computing – what is missing today is the collaboration, the exchange of sensor information and any standardised semantics for service developers. Let's see, if the next twelve years can overcome those last barriers!

1.1 Context Aware Services and Context Inference

Mark Weiser, the 'creator' of Ubiquitous Computing, had envisioned in 1995 that "in the 21st century the technology revolution [would] move into the everyday, the small and the invisible" [47]. Computing is meant to support in all situations of life, critical or not, easing people's tasks with as little need of interaction as possible. Abowd and Schilit in [1] identified the following important components for this computing paradigm:

- Scalable Interfaces,
- Ubiquitous software services,
- Ubiquitous information,
- Support for Automated Capture and Access,
- Context-aware computing and technology

The latter is responsible that "the right service for the right person at the right time and the right place" [39] is always executed; for instance that your mobile phone automatically shows the directions to the closest supermarket on your way home from work when the smart home's refrigerator senses the need for fresh milk.

Context has to be used in service selection (filtering, ranking, etc.), service management (configuration, deployment, handover, etc.), during service execution, and for proactive

service invocation. It allows for choosing among all world wide available services the one that fits best to your preferences, your profile and your current situation. It encompasses all factors that are relevant for a decision you would make or which the ubiquitous computing system should make for you.

The context for a service execution thereby not only depends on yourself. Next to the user's personal context, it obviously also includes the *service context* and the environment [43]. The *environmental context* includes for instance temperature, time, the noise level in the room, and current interactions with others, the service context can encompass e.g. the screen resolution, the colour mode, the requirements with respect to bandwidth or other contextual scope. Your *user context* can depend on many different attributes, for example:

- your location,
- your movement,
- your availability,
- your schedule and plans,
- your mood,
- your activity,
- your currently used services.

In these examples, as well as in Figure 1.1, we can see that some context information can be sensed directly with a sensor, like temperature with a thermometer or location with a GPS receiver. Other information is inherently available in computers, like screen resolution, the currently used services or also your schedule, if you use software tools to manage it.

The most challenging information is the one that cannot be sensed directly nor is available via other direct information sources, like your mood, your availability or your current activity. It has to be inferred from other information that is available, like the schedule, the location, knowledge about your preferences and accelerations of your body that can be sensed.

Such inference offers a lot of challenges to date. Mechanisms to cope with them have to be efficient enough, so millions of users can use them at the same time, the results have to be available in the shortest delays on the mobile, often resource constrained devices, and the information usable for inference constantly varies. The approaches furthermore have to deal with uncertainty of information coming from sensors that can be inaccurate or erroneous, and the fact that the same sensed information may lead to different outcomes for different persons. While a particular football result may lead to very good mood of some persons, others will feel exactly opposite about it, whilst others remain unaffected.

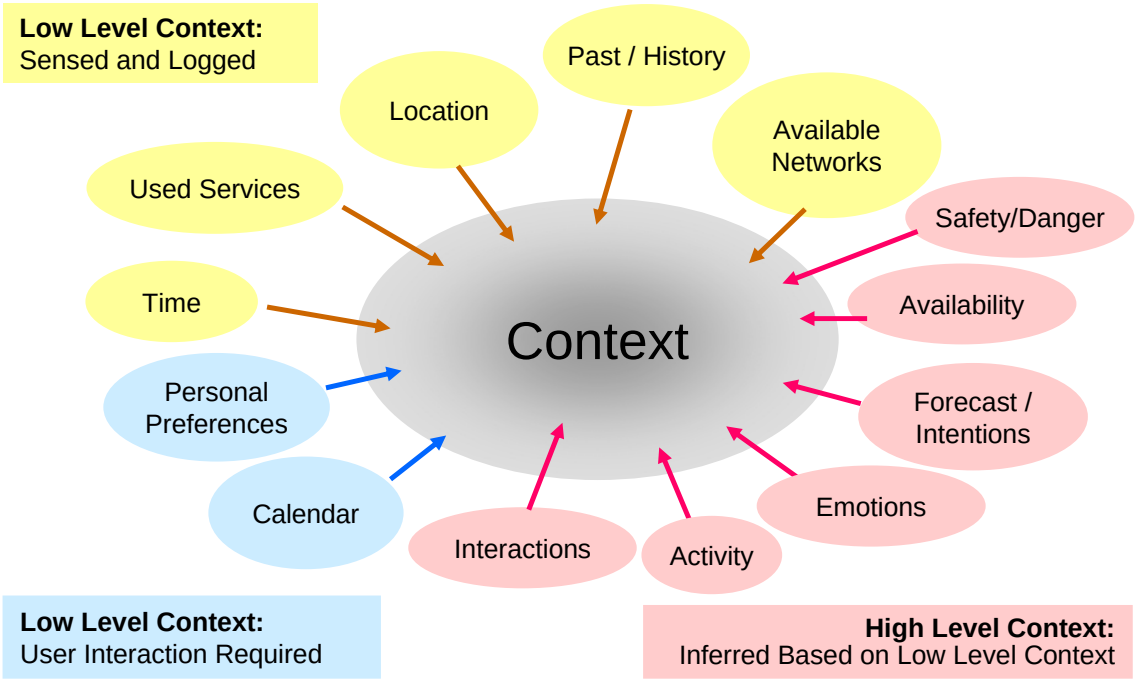


Figure 1.1: Many different factors (not exhaustively enumerated in this figure) can describe a situation and therefore form the context of a service in a ubiquitous computing environment. These factors, called *context information*, contain measurable information and things that the human user specifies like his preferences, schedules or agendas. Other context information can only be deduced, like emotions, activities or availability. This process, called *context inference*, takes into account the available context information and infers the high level context.

1.2 Scenarios

After this more theoretical explication of ubiquitous computing and context awareness, this section will give a very practical insight.

There are many different situations in which context inference is useful for next generation mobile services, ranging from Driver Assistance (DA) systems with cooperative collision avoidance over Ambient Assisted Living (AAL) for elderly people in their homes to consumer market services, including cooperative games or context aware messaging and networking services.

Three scenarios from different areas of daily life which are to be used in the following chapters for practical examples will be described and analysed regarding their usage of context awareness in the following sections.

1.2.1 An Intelligent, Context Aware Office Environment

A classical prototype environment realising Weiser's vision of ubiquitous computing is the office environment, where already many useful sensors (in smartphones, PDAs, RFID systems and many more) gather context information and mobile devices are interacted with every day.

1.2.1.1 Description

When Patrick enters his office building, a shared display in the foyer, as can be seen in Figure 1.2, welcomes him with a personalised message and informs him of the day ahead as he was recognised while passing the door. He realises that it is going to be a busy day.

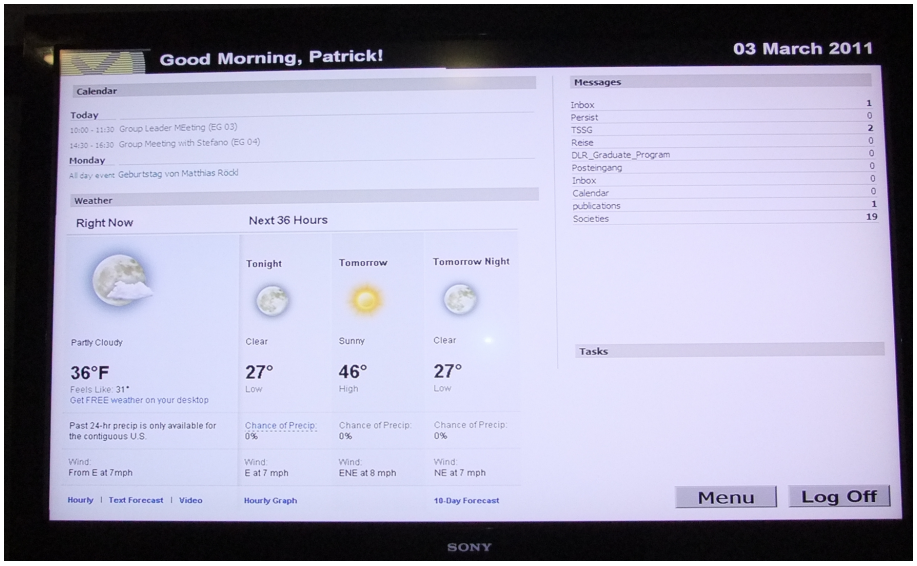


Figure 1.2: Personalised, context aware screen in the foyer of the office building with welcome message.

Entering his room, the light and climate controls adapt automatically to Patrick's favourite settings he has also defined for his smart home. As the weather is fine and warm, the windows are opened automatically to let in the fresh morning air.

Also the call redirection application is activated. This service has the following options: if a person is not in his office, a call should be redirected to the land line phone closest to him or her, whereas, while travelling or not at work, the target phone would be the mobile phone. In certain cases which are learnt and personalised from past behaviour, incoming calls are forwarded to a voice box or converted to text messages that can be accessed in more appropriate moments, like in [32]. This morning, Patrick is not in any situation in which he must not be disturbed. So all calls to Patrick are redirected from his smart phone to his office phone.

To prepare for the long day, Patrick goes to the kitchen to grab a coffee. On his way back he meets Michael in the corridor. Immediately they start an intense conversation about a research idea, Patrick had the last evening. As the building's smart space recognises this it converts one of the electronic poster frames next to them to a white board they can use for some sketches.

But Patrick has to hurry up. He has a meeting scheduled with his boss to discuss important financial and organisational issues for his research group. An incoming phone call from his yoga friend Katie is routed to his voice mail, as he is busy and the urgency level of the call was indicated by Katie to be rather low.

Right after the meeting, Patrick downloads the minutes of the meeting that have been taken automatically, as the importance of the decisions in the meeting is high, but Patrick tends to forget organisational issues. The intelligent meeting minutes service records presentations given via a projector, the contents of whiteboards, and converts speech to text. Together with the meeting agenda the key words are recognised and the minutes are structured.

After that, Patrick heads for lunch with his colleagues. As he usually wants to glance at the menu of the canteen and the weather report for the rest of the day, the wall display in the foyer now shows Patrick this information as he is walking by (see Figure 1.3).

Back in the office Patrick has, for the only time during the whole day, a longer period without scheduled meetings. He engrosses his mind in a seminal paper important for

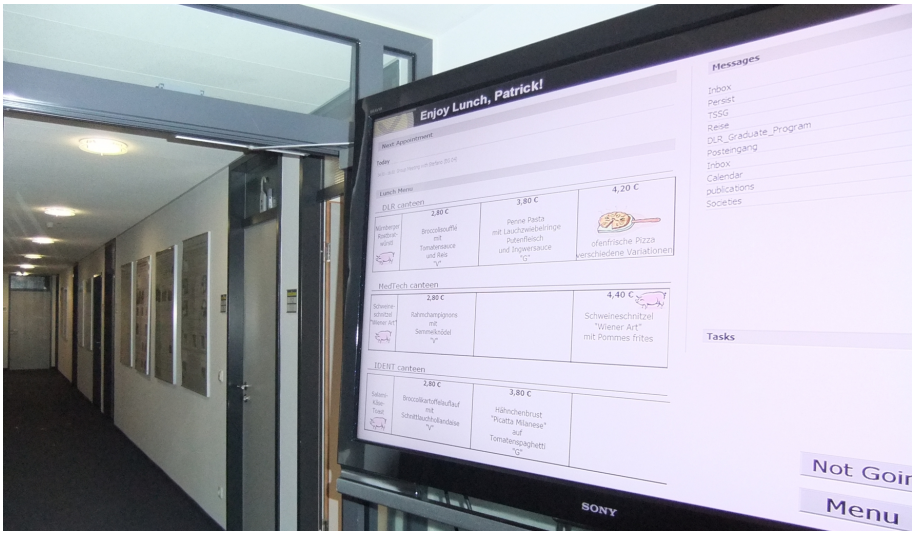


Figure 1.3: Context aware screen in the foyer of the office building when showing lunch information.

his research. Patrick's state is recognised and the smart space deduces based on his previous behaviour that he does not want to be disturbed. The call redirection service is set up accordingly and the only messages allowed to come through are those related to emergencies or the research group reunion scheduled for later this afternoon.

After two hours of fruitful work, it is exactly such a notification that stops Patrick's reflections. An alarm pops up at his screen, telling him that Stefano is entering the building. Patrick had invited Stefano, a fellow researcher from a different department of the same institute, to present his recent work in Patrick's group meeting.

Stefano is guided to the right meeting room, where Patrick is already waiting for him. When the first group members have entered the meeting room, also Michael is notified at his office PC. As he is very busy, he could not afford waiting in the meeting room. But as most meeting participants are already present, Michael also heads there – just in time to get the last free chair.

The meeting starts and after some introductory words by Patrick, Stefano is the first to present his work. He walks to the front of the room and given the agenda and behaviour, it is recognised that he is about to start the activity "presenting" and starts interaction with the wall display. He is shown a context-aware selection of suitable documents – related by last-use and association with the calendar entry – and hits the icon for the slides he prepared. The presentation is started and also logged into the automated meeting minutes.

Still impressed by Stefano's research results, Patrick accesses the presentation from the online meeting minutes and wants to print it. The respective service locates the most suitable printer on his way back to his office, sends the print job, and guides him there automatically.

1.2.1.2 Analysis

This scenario contains a number of context aware services which use context for service selection, in service execution or for a proactive service start. Some of them are continuously running in the background, like a daemon service, others are started explicitly. The following services are used:

- Foyer Wall Screen: Daemon
- Light and Climate Control: Daemon

- Call Redirection Application: Daemon
- Ubiquitous Whiteboard Service: Daemon
- Automatic meeting minutes
- Activity Monitoring: Daemon
- Guest Arrival Notification: Daemon
- Room Status Monitor: Daemon
- Projector Application: Daemon
- Printing Assistance

The context information used by these services is manifold. Among others time, location, intent, weather, availability, the caller, the schedule, interactions, meeting agendas, sound, available documents, motion, room status, high level activities and proximity are necessary to realize them.

For example in the call redirection application, running as a daemon service always in the background, activity is the most relevant context information. It depends on preferences, time, calendar entries, location, movements/status (like standing, walking, sitting and others) and the available as well as the currently used services.

The printing sub-scenario illustrates among others context aware service selection, based on features of the available printing services, but also on your current indoor position and proximity (including maps with the walking restrictions for people like walls).

Summarizing, ten services use at least 15 different contextual aspects from which many are used several times. Many of the mentioned contextual aspects again depend on other, lower level context and have to be inferred, like the motions from measurable accelerations, weather from temperature, humidity and sun intensity, proximity, the room status or the high level activities.

The daemon services need to monitor their related context and have to be reevaluated upon every context change.

1.2.2 A Context Aware Day out with Friends

The next scenario describes a day of four friends meeting in a city and spending an afternoon together while interacting with their intelligent ubiquitous computing environment, full of helpful services.

Susanna, Joan and Steve are old friends who live in or close to Dublin. They have not seen for a while and have decided to spend an afternoon together in the city. While Susanna and Joan have booked a guided tour in the National Art Gallery, Steve is still busy in the afternoon and will meet them after the tour.

1.2.2.1 Description

Joan and Susanna live outside the city and arrive at more or less the same time at the train station. Their buddy finder application informs them, so they can meet immediately. As there had been delays in the train network, they are quite late for the guided tour. Already in the train, Susanna's trip agent, keeping track of her agenda, had identified that she is late and had proposed that she went to the Art Gallery by taxi rather than by bus.

Figure 1.4: Augmented Reality museum guide in the Guggenheim Museum, Bilbao [27].

After Susanna's confirmation, the service had reserved the taxi and notified Joan. Having arrived, Susanna and Joan are guided immediately to their taxi.

The taxi's passenger comfort service is responsible for adjusting the seats, so that they are as comfortable as possible given their height and size. The taxi driver is already informed about the meeting venue, the price is agreed and paid by a secure online transaction and Joan and Susanna finally arrive on time.

Joan's friend Peter who lives in the city is already waiting there. Peter talks to Susanna for a while. This is recognised along with the fact that they have not met before. Susanna takes her smart phone from her pocket and sees the proactive prompt that just needs her confirmation: exchange electronic contacts with Peter.

The guided tour is an amazing experience. The visitors are given goggles equipped with wireless sensors. They connect to their smartphones enabling them to follow the guide's explanations in an augmented reality with supporting information displayed for the exhibits they are currently looking at, like in Figure 1.4.

When the tour ends, Steve is already waiting for them. They decide to go shopping a bit before having dinner together. They consult their social restaurant finder application. It proposes a restaurant for dinner, taking into account the location, but also weather and of course the participants' preferences regarding food and dinner time.

Having agreed on a restaurant in the city centre, Peter decides to stop by his home and to follow them later by public transport. His public transport navigation service selects the most convenient train for him to reach the restaurant and interacts with the transport management system to acquire a ticket. Finally in the train he falls asleep, as he is tired after the long tour, but his context aware alarm has recognised it and wakes him up just in time before he reaches the station where he has to get off.

Steve, Susanna and Joan are doing some window shopping as their navigation service has proposed them the route to the restaurant passing the pedestrian area and their favourite shops. Only when Steve receives an advertisement by the book shop next door that the book he is looking for is available, he enters and gets it. So they finally reach their restaurant after a nice walk with interesting chats and meet up with Peter again.

Having enjoyed a wonderful dinner, Steve offers to take Susanna and Joan to the train station. The car's navigation system guides them automatically and the built-in cooperative adaptive cruise control (CACC) [103, 46] makes for a smooth journey in the

heavy traffic. Steve is tired after the long day and he does not know the way very well. CACC exchanges messages with other cars via vehicle-to-vehicle (V2V) communications, among others regarding their position, direction, and speed, in order to avoid critical situations.

With this support, all of them have had a safe and comfortable way home.

1.2.2.2 Analysis

This scenario includes a wide range of services that use context in different ways. These services are:

- Trip Agent: Daemon
- Car Adaptation
- Buddy Finder: Daemon
- Contact Detail Exchange: Daemon
- Augmented Reality Headsets
- Social Restaurant Finder
- Public Transport Navigation Service
- Context Aware Alarm: Daemon
- Context Aware Advertisement: Daemon
- Navigation Service
- Cooperative Adaptive Cruise Control: Daemon

If we contemplate some of these services more in detail, first the car adaptation system is of interest. It starts proactively when the passengers enter and runs then only once without further monitoring further context. It incorporates the persons' preferences and context during its execution. The relevant context information is height, size and also weight of the passenger, given the same information of all other passengers. Obviously also the precise location of the passenger is needed, even within the car. Finally the user's intention to enter the car – resulting from location, movement, future calendar entries and the taxi booking – is responsible for starting the service.

The restaurant finder service furthermore includes weather and temperature information next to personal preferences, available time and availability of tables in the restaurant.

Finally the context aware navigation service or the context aware alarm have to be aware of the current time, the user's target, upcoming calendar entries as well as the map of the relevant area. Monitoring the current activity (walking-direction, but also sleeping) and precise location, it calculates the best route to the target, respectively initiates the alarm.

Similarly to the previous scenario, the used context is the own outdoor and indoor position and those of all friends, the own and the others' schedules, the trip destination and time along with the available means of transport and their schedules, if applicable. Height, weight and sitting preferences, current interactions with other people (and consequently their activity), the direction somebody is oriented towards, preferences regarding time and style of dinner and those of the other friends, as well as the own shopping list are

also incorporated. For CACC, position, direction and speed of other cars are taken into account along with the own one, but also weather, pavement condition, traffic density etc. can be important.

1.2.3 Smart Homes and Ambient Assisted Living

How the improvements of context aware ubiquitous computing can also improve life at home is shown in the following scenarios.

William, aged 78, lives alone in his own house. He is managing his activities of daily living sufficiently well, in particular with the help of his smart home that eases many tasks, see Figure 1.5. In his smart home he accepts some monitoring of his daily routine, which allows for further independent living as opposed to moving into a nursing home.

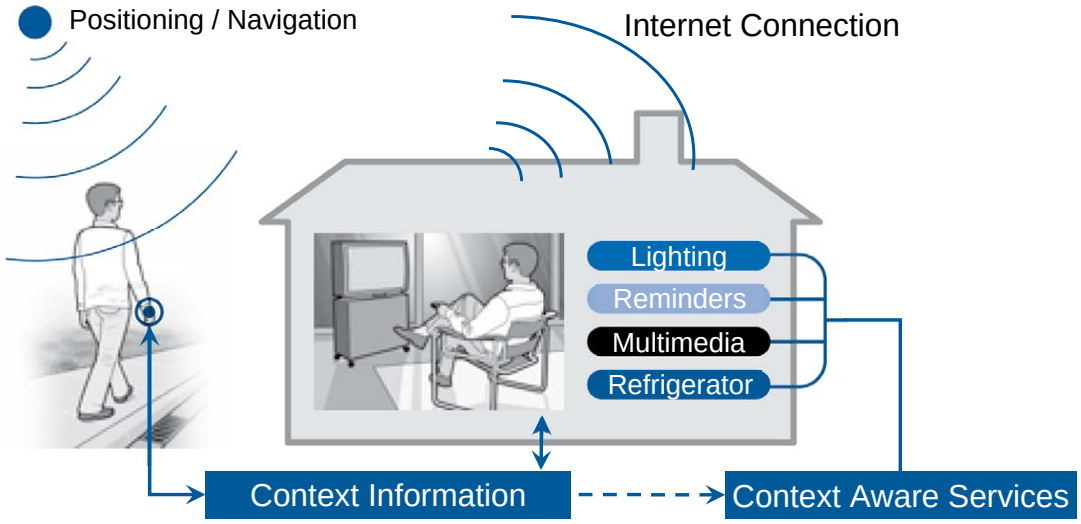


Figure 1.5: A context aware environment combining a smart home and outdoor services, adapted from [6].

1.2.3.1 Description

William wakes up in the morning. While still lying in bed, his Voice Prompt Reminder application has stated twice in the last 15 minutes that he is overdue for taking his medication. He knows that it is very important to take his medication regularly and so gets up to take it.

Unluckily, William has lost one of his slippers under the bed. He gets on the ground to search under the bed. The smart home's monitoring system knows it is William who is lying on the floor, but using the available sensor data such as motion sensors on William's body, the system recognises that he has not fallen to the ground, but is lying on the ground intentionally with 85 % certainty. His daughter living in the same town will be only notified if the certainty threshold for a fall goes above 30 %. As he gets up again and moves normally, no alarm is sent to his family or caregiving service.

After breakfast, the next reminder tells William that he had intended to go to the supermarket and to collect his clothes from the laundry. William takes his mobile interaction device and confirms that he is going to do that now. The device proposes the shortest and safest route combining both targets.

When William enters the supermarket, he is shown his shopping list. It was mainly populated automatically by the refrigerator that notices and logs which products have been

put in and taken out. The intelligent shopping list starts to interact with the supermarket's smart space and William is guided automatically to the products he is looking for. Having collected his clothes he returns home.

Now William starts preparing his lunch. A cookbook service proposes him a suitable recipe, taking into account his diet, his lunch preferences and also the available ingredients. With the accompanying guidelines, the dish is easily prepared and delicious. When William takes the pan from the oven, the latter is deactivated automatically, to avoid unnecessary energy consumption as well as possible hazards.

In the afternoon, William is reminded to do his daily gymnastics to maintain his health. His sports plan is adapted thereby automatically to the number of calories he has consumed and the amount of exercise he has had already during the day, monitored by his movement sensors. As William had already gone to the supermarket, he is proposed only twenty minutes, mainly focussed on stretching and balance exercises. If William did not fulfil his sports program for a couple of days or his motion patterns show abnormalities, his daughter would be notified.

Before going to bed, William wants to see a film on TV. He sits down in his armchair and takes out his mobile interaction device. This device infers from both, the current situation and the pointing direction of the device, that it should switch to the TV remote control mode, while adapting the light settings to the movie profile.

1.2.3.2 Analysis

In this scenario the following context aware services are used:

- Voice Prompt Reminder: Daemon
- Home Monitoring System: Daemon
- Fall Alarm: Daemon
- Route Planer
- Smart Refrigerator: Daemon
- Intelligent Shopping List
- Supermarket Indoor Navigation Service
- Cookbook service
- Automatic Oven Deactivator: Daemon
- Sports Planer: Daemon
- Movement Logger: Daemon
- Health Monitoring and Notification System: Daemon
- Context Aware Remote Control: Daemon
- Context Aware Light Management: Daemon

Most of these services are continuously running in the background as daemons. The smart home itself can have built-in servers, cables and a permanent power connection; – therefore it is not a resource constrained device. The mobile interaction device however that can even serve as remote control for all devices in the living room, has to infer some values itself, in particular when it is taken outside the smart home. Then limited energy, network bandwidth and computing power need to be considered.

The applications take into account time, the current activity, task lists, the room status, movement of persons, location, the contents of fridge and cupboards, available recipes, position of goods, the status of electrical devices, consumed calories, diets, the history of physical activities, health benchmarks and personal history, available devices, light, direction of the interaction device and ambient sound.

Also, here we can observe that some services use context to get started, others during execution, as well as context aware selection of such services, when multiple instances of the same service type exist. These different usages entail different monitoring schemes for the respective context.

Also here we can notice that much information has to be inferred from multiple sources of lower level information.

1.2.4 Summary

The scenarios shown in the last sections show how context can be used in the future. Many different services are running continuously and in parallel on mobile, resource constrained devices using context information which is not directly measurable.

This thesis will propose to use Bayesian methods to infer the necessary context information from the available measurements considering the specific requirements of ubiquitous computing.

1.3 Objective and Research Questions

Over the last 15 years, *Bayesian Networks (BN)* have evolved as a major tool in a wide area of scientific disciplines requiring sound statistical analysis, automated reasoning or exploitation of knowledge hidden in noisy data. These range from medical research, genetics, insurance analysis, and fault handling to automation and intelligent user interaction systems. BNs combine techniques from graphical models with those from Bayesian analysis to provide a formal framework where complex systems and uncertainty can be represented and analysed. As such, they are also a candidate for context inference.

Although the concepts of context aware services and also Bayesian networks as means for inference have been known for more than fifteen years, context aware services (CAS) are still not part of daily life. One reason is the availability of sensors that has only recently begun to increase due to sinking hardware costs. Only few common standards are agreed, which delays large scale deployment. In the author's opinion, a major problem however is also context inference. Existing algorithms are tightly coupled to specific sensors and context aspects, application developers cannot simply add inference rules, and all-encompassing approaches would not scale and not be tractable for mobile devices.

This work proposes Bayesian techniques as the main building block for context inference systems, with context attributes represented as random variables in causal BNs representing inference rules. This work proposes a way to fit such methods into context management systems that is computationally tractable for resource limited mobile devices and scales for large numbers of users. An important factor for this is the reflection that only the necessary information must be involved in computations. Therefore this

work proposes ways to dynamically adapt inference to the current situation and the user's personal preferences.

The most crucial challenges this research faces can be summarised in the following questions:

1. How can context inference with BNs be made tractable for resource limited mobile devices?
2. How can BN inference rules be adapted to the changing mobile environments?
3. How can a ubiquitous computing system personalise context inference?
4. How can context inference with BNs be integrated in a standard context management system?
5. How should a system learn and adapt to changing needs or preferences causing a different situation estimation?
6. How can this system of inference rules be designed to be extensible and pluggable, so always the relevant information is evaluated, only?
7. How can the most important influence factors of context be recognised most efficiently?
8. How can different inference requirements for different contextual aspects be fulfilled and how can different inference methods interoperate?
9. Are Bayesian methods equally suitable for low level and high level context inference?
10. What are the advantages of the methodology proposed in this work?

1.4 Thesis Outline

To answer these questions, after a thorough analysis of the research which already exists in chapter 2–4, chapter 5 will present the overall concept proposed in this work. As can be seen in Figure 1.6 its is applicable to low level as well as to high level context inference and determines the relevant aspects of the situation which form the service context.

Chapter 2 will explain the definitions and theory underlying this work. It gives an overview over Ubiquitous computing and context awareness, as a key to a context aware system is sensing, identifying and using context information and making it accessible to all context consumers, i.e. services. Current methods for inference are presented, encompassing approaches based on logics, data mining, artificial neural networks, and probabilistic methods. In this thesis it will be argued that the latter are most promising for our domain, in particular Bayesian methods that build on conditional probabilities. Therefore a specific focus of this chapter is the theory of Bayesian and in particular causal networks, an interpretation of BNs that eases modelling the dependencies between different sensors and inference targets.

Since one major problem of today's context aware frameworks is scalability, chapter 3 illuminates the target user domain, based on assumptions and observations about the number of service users, their service usage and context dependence of services. This chapter will show that today's all-encompassing, often central approaches would not be able to handle such a load. From these requirements, a viable approach can be extracted, like decentralisation of inference, which entails the need for modularity and privacy control.

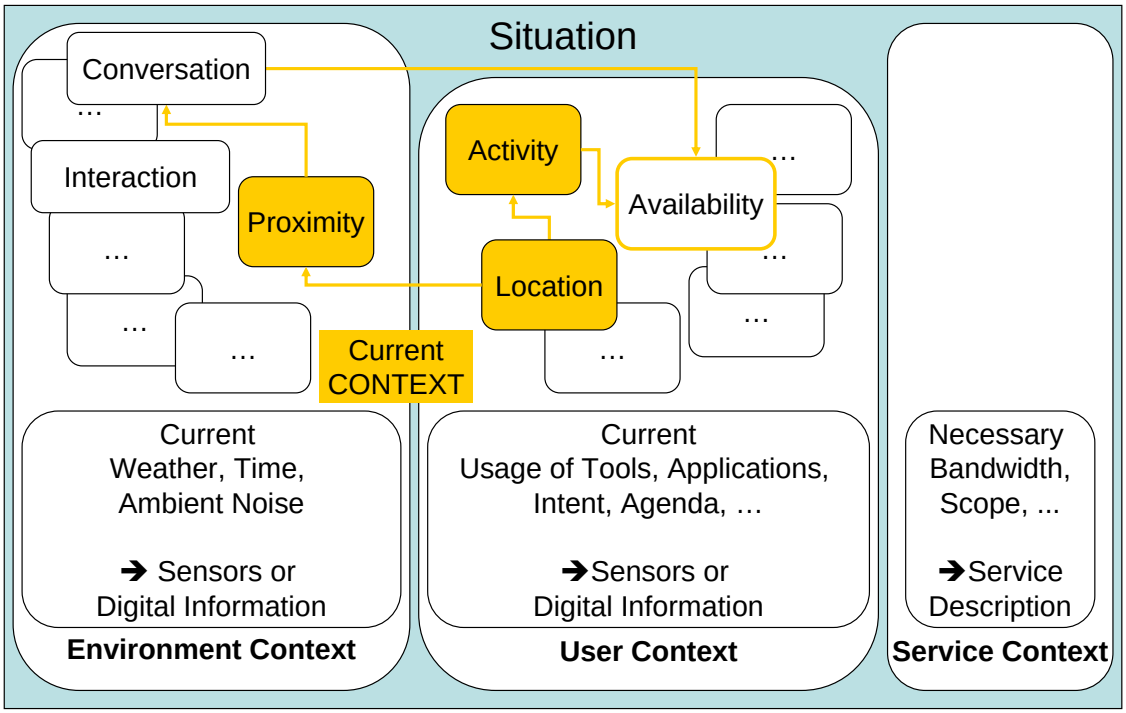


Figure 1.6: This thesis covers a Bayesian context inference approach applicable to both, lower and high level context describing the user himself, and his environment. It furthermore proposes means to identify and relate context aspects to that part of the situation which is relevant for a service execution. The orange elements in the figure indicate main topics of this work, i.e. activity, location, proximity, availability and the connections of context aspects.

Chapter 4 presents the specific state of the art that this work affects. The inference of high level context can involve numerous aspects of lower level context. As it could be seen in the scenarios, the two central ones that influence most high level context are (i) human motion related activity and (ii) location. The state of the art in recognising both is presented here along with existing research about structuring and modularising BNs and inference in such networks. Although promising approaches exist, they would not be sufficient to realise scalable and adaptive high level context inference.

The necessary improvements over the state of the art are presented in chapter 5. A first section explains how the proposed techniques are employed for context inference. Then another section proposes Bayesian approaches for human motion activity and position recognition that can be integrated with a Bayesian context inference architecture, is scalable and resource efficient. Once the reader has seen how lower level context is inferred, the next step is the identification of those contextual aspects that are most relevant for a given inference goal. The fourth section of this chapter will present an approach for modular context inference which reduces inference costs compared to normal BN inference. Subsequently, ways to automatically assemble inference modules and to infer across different modules are explained.

To prove that the concepts are viable, all components were implemented in a Java based framework. In chapter 6 the application of the presented concepts will be demonstrated and evaluated, referring back to the analysis of the current situation from chapter 3. Chapter 7 finally closes this work with a summary and conclusions.

Chapter 2

Fundamentals of Context Inference

This chapter presents the fundamentals of this work. It includes basic definitions and research on ubiquitous computing that are necessary for this work, as well as the probabilistic theory and techniques for the Bayesian methods used herein for context inference.

2.1 Context in Ubiquitous Computing

"Ubiquitous Computing" is said to be invented by Mark Weiser who coined the term in 1988 while working for the Xerox Palo Alto Research Center. He envisioned a world with hundreds of different sized computers and sensors per room, wireless access to a global information network and seamless integration of computing into daily life with intuitive user interfaces [47]. In this work it will be used synonymously with "Pervasive Computing".

One of the keys to realising ubiquitous computing is context awareness. The following sections shall present the basics of context awareness and context management.

2.1.1 Definitions

As a first step to this, some important definitions from the literature are discussed which will be used in the remainder of this work.

A widely used definition of context is the one given by Dey in [9]:

> Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.

This definition has been developed further by Strang and Linnhoff-Popien in [40]. They propose the definitions 1 – 6 distinguishing clearer between "situation" and "context", specifying "relevancy" and introducing an interesting concept, the "aspect". The present thesis follows this set of definitions.

Definition 1 *Context Information*:
A context information is any information which can be used to characterize the state of an entity concerning a specific aspect.

Definition 2 *Entity*:
An entity is a person a place or in general an object.

Definition 3 *Aspect*:
An aspect is a classification, symbol or value-range, whose subsets are a superset of all reachable states.

Definition 4 *Situation*:
A situation is the set of all known context information.

Definition 5 *Context*:
A context is the set of all context information characterizing the entities relevant for a specific task in their relevant aspects.

Definition 6 *Relevancy*:
An entity is relevant for a specific task, if its state is characterized by at least concerning one relevant aspect. An aspect is relevant if the state with respect to this aspect is accessed during a specific task or the state has any kind of influence on the task.

If a service execution framework supports context aware services it hence needs the functionality to give the service access to context, i.e. the relevant context information. The provider of such functionality is called *Context Management System* (CMS). Following the argumentation of the author et al. in [118], it can be defined as follows:

Definition 7 *Context Management System*:
A CMS is a distributed middleware that provides users and services access to context information upon request and/or subscription, if the requester has been granted the respective access rights. It is based on an efficient, semantically defined context model, manages context sources and a history of context. Furthermore it models, manages and exploits quality of context and provides facilities for reasoning and context inference.

The literature uses again many different definitions and understandings of reasoning and context inference. This work distinguishes them in the following way:

Definition 8 *Context Inference*:
Context Inference creates new context information from available information. It thereby copes with imperfection and uncertainty of information and the derivation of higher level context.

Definition 9 *Reasoning*:
Reasoning is a wide set of methods enabling the relation of context information, definition of procedures, rule and case based decisions for context information, as well as context inference.

This distinction follows to some extent Perttunen, Riekki and Lassila in [132] who call "activity and context recognition" what this work names "context inference". They, however, do not see this as part of "reasoning", in contrast to Bikakis et al. in [110] or Bettini et al. in [109].

Definition 10 *Context Inference Rule*:
Context inference rules (CIR) are models used by machine learning algorithms to produce new context information in the process of context inference.

Furthermore, the concept of ontologies is important for context awareness. Gruber in [17] has defined an ontology to be a "specification of a conceptualisation". Uschold in [45] is more concrete: "An ontology may take a variety of forms, but necessarily it will include a vocabulary of terms and some specification of their meaning. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms". The following definition specifies the latter:

Definition 11 *Context Ontology*:
A context ontology is an ontology defining the basic meta vocabulary for context usage. Moreover it specifies, not necessarily exhaustively, the usable context and its interrelations.

2.1.2 Context Models

The representation of context and the models for it have been one of the first context related research fields. They are not only related to the nature of the context information, but also to the needs of context consumers and tightly coupled with the inference that builds on them [121, 132].

Strang distinguishes in [41] six different categories of context models. Key-value models, markup scheme models, graphical models, object oriented models, logic based models and ontological models are rated according to the requirements of *distributed composition*, *partial validation*, *richness and quality of information*, *incompleteness and ambiguity*, *level of formality*, and *applicability to existing environments*.

Most of these models however are relatively equivalent – ontological models can be represented as sets of keys and values, graphical models always can be described in ontologies, ontologies are based on logics. The key for their usefulness is rather the nature of the meta information, as the meta information is used in reasoning and inference. While inheritance relations are of high importance for structural reasoning, the likelihood of a value and the probability distribution of the random variable are crucial for Bayesian methods.

Perttunen, Riekki and Lassila in [132] have imposed the requirements of *unique identifiers*, *validation*, *expressiveness*, *uncertainty and incomplete information*, *simplicity, reuse and expandability*, and *generality*.

In particular they conclude that finding the trade-off between efficiency, expressiveness, soundness, and completeness has not been studied so far, that the problem of relevance is left for the applications using context and that benefit of modelling uncertainty and vagueness has not been evaluated beyond the capability of representing, i.e. that no application has shown its utilisation to date. Stating this, they neglect however that the representation of uncertainty and vagueness is not only of use for context end consumers, but also for context inference.

The model underlying this work is the "Aspect, Scale, Context Information (ASCi) Model" defined by Strang, Linnhoff-Popien and the author in [42] building on the context (Definition 5) and in particular aspect definitions given above in Definition 3.

Context information (see Figure 2.1) always belongs to a specific *scale* or "*type*", these scales and types in turn are specific to uniquely one contextual *aspect*. Context information is related to at least one entity and specified further by meta information. Meta information represents the *quality of context* (cf. [8]) and contains at least the timestamp of its creation and every further possibly necessary information.

Like that it can represent all relations necessary for reasoning and in some meta information also the likelihood or the type's probability distribution necessary for Bayesian

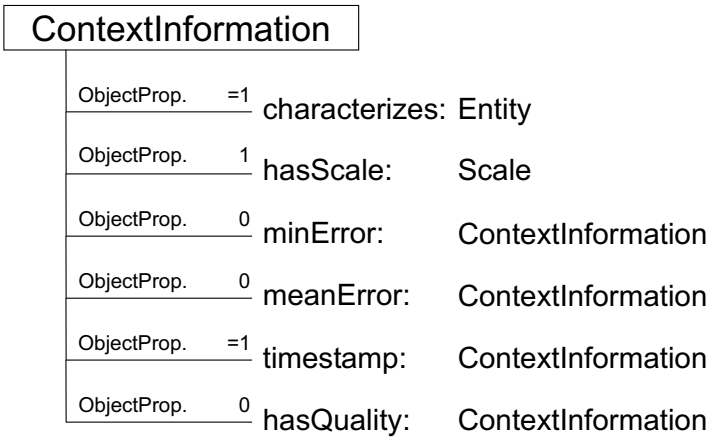


Figure 2.1: Context Information Model following the ASCi Model defined in [42].

context inference methods. Context information can be represented graphically, as RDF[1] triples, in OWL[2] or DAML+OIL[3], as well as with relational models in a database.

2.1.3 Context Management Systems

Research activities worldwide have produced first prototype CMSs, most offering a basic but not generally applicable interface for context inference. A collection of such frameworks has been surveyed by the author et al. in [118]:

One of the first context management systems capable of handling generic context information was developed for the needs of the Cooltown project by HP labs in 2002. The Cooltown context management system attempted to resolve problems regarding the context representation, while it combined and exploited context information by introducing a uniform web presence model for people, places and things [26]. Cooltown envisioned a world where "humans are mobile, devices and services are federated and context-aware and everything has a web presence". Various prototypes have been developed based on this perception, the most important of which are: museum exhibits that interact with the user, conference rooms that recognize users and automatically adapt to their presence, and radios that play songs based on the preferences of users in the proximity. However the provided context management system did not handle issues regarding context history, inference of context data, quality of context, context privacy and security.

Owl [12] is a context-aware system aiming to gather, maintain and supply context information to clients, while protecting people's privacy through the use of a role-based access control mechanism. Apart from the provision of basic context related functionalities, the Owl project performs an initial research on more complex issues such as history of context, access rights, quality, extensibility and scalability of context information [19].

Another project that provided an architecture for the provision of mobile context-aware services is *SOCAM (Service-Oriented Context-Aware Middleware)*, presented by Gu et al. in [18] (2004). This architecture models context information around four main context concepts: person, location, activity and computational entity (e.g. device, network, application, service, etc.). The SOCAM context model is specified in OWL and addresses

[1] Resource Description Framework, see `http://www.w3.org/RDF/`
[2] `http://www.w3.org/TR/owl-features/`
[3] `http://www.daml.org/2001/03/daml+oil-index.html`

context sharing, reasoning and knowledge reusing, while providing a service oriented middleware infrastructure for indoor applications, where a central server retrieves context data from distributed context providers and delivers them to its clients after proper processing.

The DYNAMOS project from 2007 [36] aims at providing mobile users with context-aware services, focusing on proactively notifying them about services they are possibly interested in. The main context information used is the user's personal profile, which is a combination of its personal information, preferences and schedule, including the user's activities and a personal calendar. This information is stored and can be shared with other users, rendering the sharing of services possible based on privacy criteria set by the users themselves. Context management functionality is provided by the Contory [35] middleware from 2006 which is specifically designed for resource-constrained devices, such as smart phones.

The *CroCo (Ontology-Based, Cross-Application Context Management)* [33] context management service (from 2008) aims to support domain independent applications by handling arbitrary context data, provided by context providers and requested by consumers via a service interface. This is achieved by adapting an extensible context ontology allowing the integration of external ontologies describing contextual aspects relevant for various domains. Among others, the basic design principles of the CroCo architecture include consistency maintenance of the distributed data and reasoning about context information.

For the needs of the IST DAIDALOS (2005) and IST DAIDALOS II[4] (2008) European research projects, the *CDDBMS (Context Distributed Database Management System)* [34] has been designed and developed. The CDDBMS is a distributed heterogeneous multi-database system that is built to face the requirements of network operators and context marketplaces, while being scalable and lightweight resembling a web-server schema. It also provides basic interfaces to context inference, query extension mechanisms and free-text based query handling, context access control mechanisms and for managing context history.

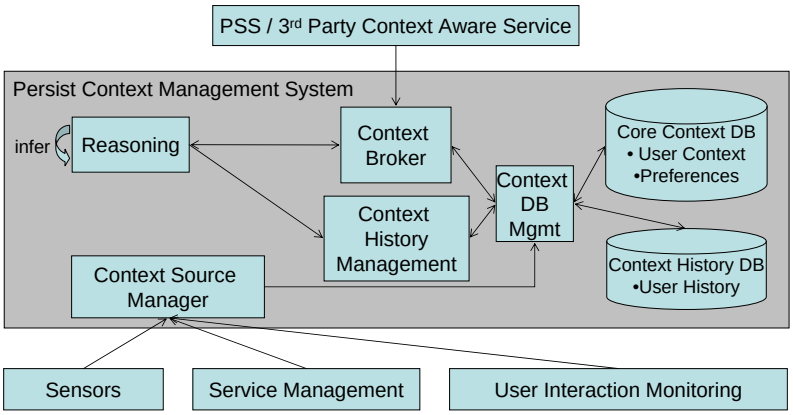


Figure 2.2: The PERSIST CMS: It consists of five architecture blocks. The Context Broker is the core interface to any consumer, the Context Source Manager is the only access point for all context sources and sensors. Both components store their information in the Context Database via the Context DB Manager, the central information unit. The Context History Management provides the necessary, pre-processed information e.g. for learning inference rules, the Reasoning finally is the place where inference algorithms are stored and processed whenever necessary.

[4] `http://www.ist-daidalos.org/`

The ICT PERSIST[5] project has advanced the DAIDALOS CMS with a focus on context management for personal smart spaces, resulting in the architecture shown in Figure 2.2. In particular it already uses quality of context information to some extent, provides a context history database for finding repetitive patterns and a distributed management system for consistent access to context described by Roussaki et al. in [134]. It provides basic reasoning and context inference capabilities, like the estimation of proximity, Bayesian high level context location and physical activity [120].

The Persist CMS serves as prototype for the context management in the present research. With the focus on integrated and better scalable context inference facilities, less effort has been put in distributed management and history management, however.

2.2 Context Inference

As defined in Definition 8, context inference generates new knowledge from existing information. It therefore evaluates models which relate this existing information (low level context) and the desired target information (the high level context). This leads to a classical classification problem, i.e. checking which "class" (i.e. which state of the high level context) best fits the combination of low level information. The models for the classifier can either be set manually or learnt from recorded data, are usually specific to one inference or classification approach, and impose requirements on the modelling of the low level (input) context information.

Methods used for inference should be efficient, sound and complete according to Perttunen et al. in [132] and cope with imperfection of data [110]. Angermann et al. [107] specified the requirements further, suggesting means for fusion of several input sources and handling of possibly contradicting measurements, expressive modelling of situations, and adaptation to the needs of large scale pervasive systems, i.e. distributed processing, personalisation and adaptability to the system's dynamics.

Different authors (e.g. [121, 109, 110, 132, 129]) have identified different categories for inference approaches. Harmonising these works, methods for context inference can be grouped into (i) logical approaches with many different subforms, (ii) neural network based inference, (iii) kernel machines, (iv) instance based approaches and (v) probabilistic context inference. The next paragraphs will give a short overview of the approaches, their utilisation in the literature, and their advantages and drawbacks.

2.2.1 Logic Based Approaches

Context inference with logic based approaches is the process of deduction of knowledge from the knowledge base. It is by far the most frequently applied approach for reasoning in today's context aware systems [110]. There are many different logical languages with different operators, properties and inference options. The remainder of this section shall give a short overview.

Propositional Logic is the basic form of logic. It consists of atomic sentences consisting only of a single proposition symbol, and complex sentences using one of five logical connectives. The connectives are *negation* ($\neg$), *conjunction* ($\wedge$), *disjunction* ($\vee$), *implication* ($\rightarrow$), and *biconditional* ($\leftrightarrow$) [66]. It can be shown that sets of two connectives,

[5] `http://www.ict-persist.eu/`

$(\neg, \wedge)$, $(\neg, \vee)$, $(\neg, \rightarrow)$, are sufficient to represent all other connectives. These sets are called *functionally complete*.

First Order Logic extends propositional logic. While the latter models the world as boolean composition of *facts*, first order logic assumes that the world also contains further elements, such as objects, relations and functions [66]. It allows for quantors and multiple instances of facts.

As a real superset of propositional logic, it increases expressiveness, but sacrifices decidability. I.e. for general first order logic there is no algorithm that can decide whether arbitrary formulas are logically valid. This is a serious restriction for classification.

Description Logics (DLs) are proper subsets of First Order Logic where some expressiveness has been sacrificed in order to make the problem decidable. They provide formal languages for constructing and combining category definitions and efficient algorithms for subsumption and consistency tests as well as classification [66].

Higher Order Logic extends first order logic by seeing relations and functions as objects in themselves and in this way allows assertions about all relations. It is a proper superset of first order logic [66]. With their extended semantics they are more expressive, but their model-theoretic properties are less well-behaved than those of first-order logic, complicating classification even more.

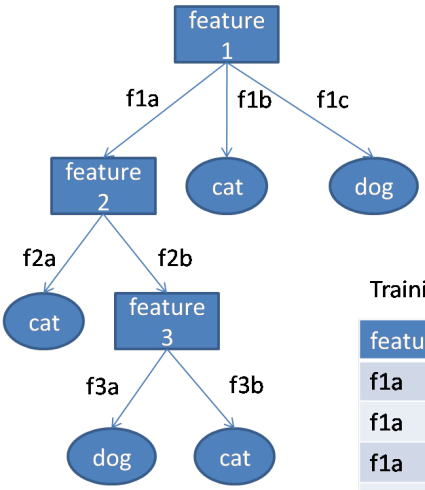
Temporal Logic assumes that facts hold true at particular moments or intervals in time which are ordered [66]. It introduces new connectives with regards to propositional logic, *Until*, *Release*, *Next*, *Future*, *Globally*, *Exist*, and *All*. It can be used to represent statements that hold only for certain time intervals [133].

Inference with logics is possible in many different ways. Particularly widespread is the classification with *Decision Trees* building on propositional logic. The leaves of the tree represent the classes, the branches to these leaves contain nodes, called "features". For context inference, the different values of the target high level context represent the classes i.e. the leaves, the values of the features, every feature represents one of the low level context types. Their values decide which branch to follow, starting at the root of the decision tree, see Figure 2.3.

A data set allows learning of different decision trees. The goal thereby is to build a minimal tree using those features as close to the tree's root as possible which discriminate the different classes best. Different methods and parameters are defined in order to identify the quality of discrimination [129]. A big problem is that learnt decision trees from too sparse data sets might over-fit data. The performance of decision trees decreases if the instance space is not orthogonal to the axis of one variable and parallel to all other axes conforming hyperrectangles [129].

Inference in first order logic can be grouped into three families: *forward chaining*, *backward chaining*, and *theory proving systems*, see [66] for details.

In context inference, literature uses logical inference in particular in combination with ontologies that represent context models and information. Ontology based context inference uses for example First-Order Logic and Temporal Reasoning in CONON [138]), but mainly Description Logics [132]. For instance [115, 126] provide deductive reasoning to extract previously described knowledge from large knowledge bases. Those approaches build the vast majority of all context inference systems to date, but in fact they cannot



Training Set:

feature 1	feature 2	feature 3	class
f1a	f2a	f3a	cat
f1a	f2b	f3a	dog
f1a	f2b	f3b	cat
f1b	f2a	f3b	cat
f1b	f2b	f3b	cat
f1c	f2b	f3a	dog
f1c	f2b	f3b	dog

Figure 2.3: An example of a decision tree from [237] with three features and two classes. It classifies *cats* and *dogs* based on the input data given by the training set shown on the right.

create new high-level context like users' activity or situation: there are large problems in describing causal (but not strictly deterministic) relations between concepts, and mainly they cannot cope with uncertain or missing information. Logic subsets that are more expressive with regards to knowledge description on the other hand lead to non-decidable problems for reasoners. Scalability with ontological, description logic based inference moreover seems questionable [132]. Kleek and Shrobe in [127] give an approximate number. They collected about 300 MB of RDF represented data for one person in three weeks. Processing and storing such large amounts of data will become practically infeasible in decent response times. Inference complexity is NP complete, as it can be reduced to the boolean satisfiability problem (*SAT-Problem*) [81].

Fuzzy Logic is a logical formalism trying to incorporate uncertainty about a proposition. Beyond *true* or $false$, propositions can have any continuous value between 0 and 1 so that the propositions become *fuzzy sets*. "Fuzzy sets are an instrument of modelling inexact predicates appearing in natural languages" [131]. The function, assigning a value to a proposition is called *membership function*, where 0 represents "no membership at all", hence *false*, and 1 *true*. The available operators are conjunction, disjunction, negation and *modification*.

Fuzzy logic however does not take into account the correlations or anticorrelations of propositions [66]. For instance the membership of a proposition and its own negation can be evaluated to a value that is not 0.

2.2.2 Neural Networks

Artificial *Neural Networks* are supposed to simulate a brain's information processing capacity. They are composed of nodes, representing neurons, and directed, weighted links [125]. Every node is associated an activation function which fires, if the sum of the weights of its inputs exceeds a threshold, see Figure 2.4.

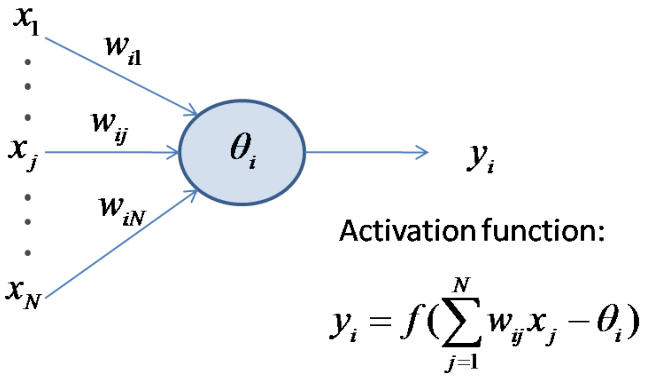


Figure 2.4: Model of a neuron for an artificial neural network taken from [237]. It receives the input signals x_i weighted by the synaptic weights w_{ij} and applies an activation function to the input signals considering a threshold θ_i. The output signal y_i of the neuron can be the input of other neurons of the network.

There are two main categories of neural network structures [66]:

1. acyclic or *feed-forward networks*
2. cyclic or *recurrent networks*

Feed forward networks represent a function of its current input without an internal state. A recurrent network feeds its outputs back into its own inputs, thereby realising an internal state or memory. Feed forward networks can have several layers, designed such that each neuron receives input only from neurons in the immediately underlying layer.

In 1958, Frank Rosenblant developed a single layer feed-forward neural network, called *Perceptron*. Well-known algorithms are based on the notion of perceptrons. A perceptron assigns each input an output following a learning process of error correction in order to determine the correct weights. Using the lower level context information as input signals with given weights w_{ij}, the neuron computes $\sum x_j w_{ij}$ and compares the sum with threshold θ_i [129]. If the sum is above threshold, output is 1, else it is -1.

Both, single-layered and multi-layered perceptron neural networks are used in classification. They can represent complex non-linear functions with many parameters and can be learnt from noisy data. A single layer perceptron does not have any hidden variable, can represent only linear separable functions and a simple weight update rule can be used to fit the data exactly. Multilayer feed-forward neural networks can represent any function, given enough neurons.

Comparison of decision trees and neural networks [129] has shown that neural networks usually perform as well as decision trees, but hardly better. The main disadvantage of neural networks is the difficult and time consuming way to train them, as well as the fact that they cannot be set manually or interpreted – they can only be seen and used as a "black box". In addition, the layered structure of the network has to be defined by human users for learning. As such they are not an ideal solution for context inference.

2.2.3 Kernel Machines

Kernel machines are techniques to cluster data in a relatively efficient model learning process, able to use complex, non-linear functions as discriminator between different classes. They find hyperplanes that linearly separate classes with a maximum margin, after applying a kernel function to the dimensions as in Figure 2.5. As such hyperplanes can be

defined by the vectors defining them, the support vectors, Kernel Machines are also called *Support Vector Machines* (SVMs).

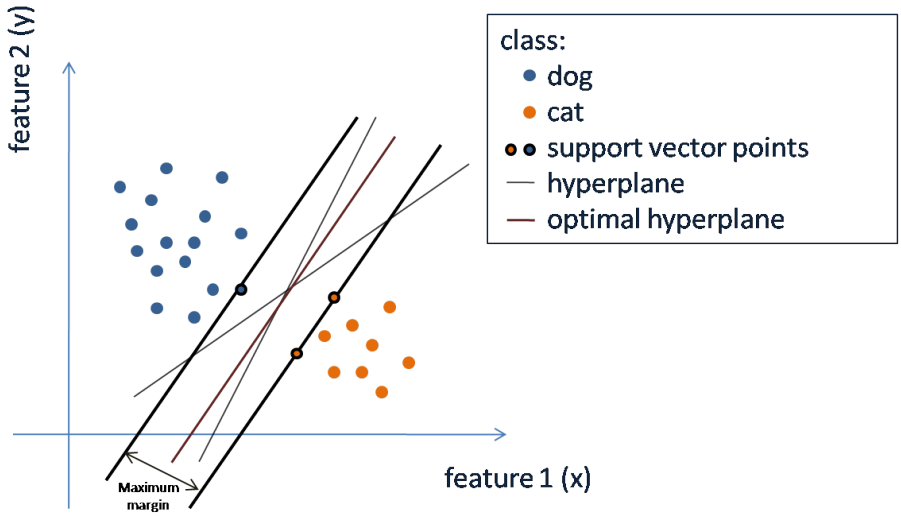


Figure 2.5: Support vector machines try to find the maximum margin hyperplane that separates the different classes in a recorded data set. Support vector points lie on its margin and are shown as well. Figure taken from [237].

As the final solution is a linear combination of the support vector points, there is no need for a large number of training instances [66]. If the data set contains misclassified instances, a *soft margin* that accepts missclassifications of the training instances can be used [129].

SVMs are a supervised machine classification technique, resolving binary classification problems only. If more than two classes have to be distinguished, a hierarchical model has to be established that divide the problem into several binary decisions. The selection of the appropriate kernel function is not trivial, and also the usage of discrete and nominal (i.e. unsorted) values is problematic [129].

Given that the models cannot be set manually and benefit from large data sets to reach its ideal prediction accuracy [129] and that low level context information is in our case usually discrete and nominal, SVMs cannot be seen as a feasible solution for context inference.

2.2.4 Instance Based Approaches

Instance based models represent a classification constructed directly from the training data. Next to kernel methods which form a distance-weighted combination of all the instances [66], *k-Nearest-Neighbour*(kNN) models [113] are the most important representatives of this category of inference approaches.

The underlying idea is that the class of a data instance is the same as the class of data in close proximity. This classifier considers every low level information as a dimension of a n-dimensional space where every data instance (that corresponds to a set of low level information) is a point within this n-dimensional space. kNN considers the relative distance between instances. Distances can be defined in different ways, depending on the nature of the low level information, e.g. Euclidean, Minkowsky, Manhattan, Chebychev, or Hamming distance [129, 66].

For inference, the data point representing the measured low level information is assigned the class label most frequent among the k nearest instances around the input data point (see Figure 2.6).

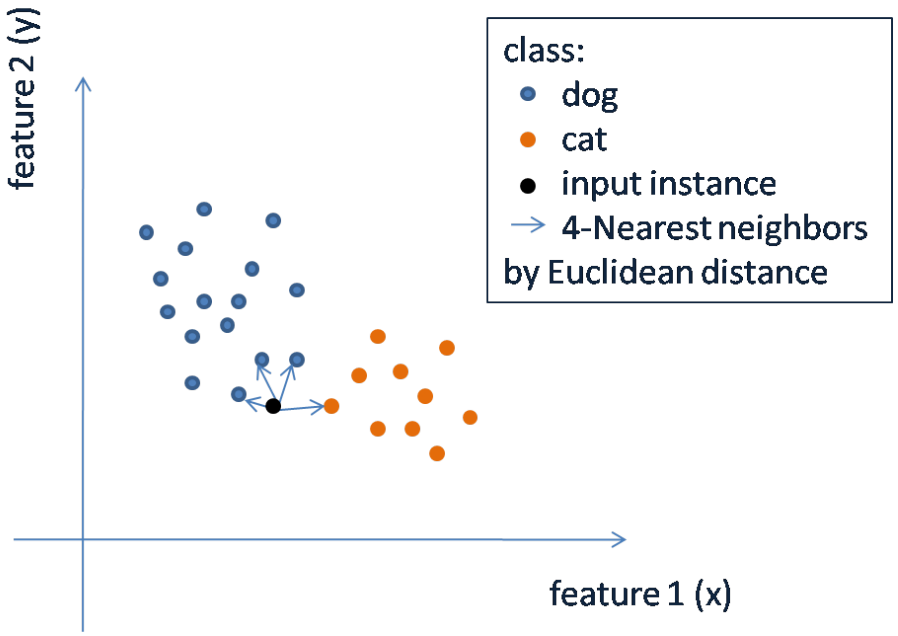


Figure 2.6: An example of application of the k-Nearest-Neighbour algorithm with two dimensions, hence two low level context information given in [237]. The classifier decides if the class is *dog* or *cat* given the input data represented as a black point. Using the Euclidean distance and for a value of $k = 4$, the class and therefore the inference result is *dog*.

Instance-based learning algorithms require less computation time during the training phase than other algorithms, such as decision trees or Bayesian Networks, but more computation time (depending on the size of the dataset) during the classification process [129] which is not appropriate for context inference. Also their storage requirements and their sensitivity to the choice of the relative distance and parameters [129] are critical. Fast classification is crucial to minimise response time in ubiquitous computing, context attributes are too heterogeneous for overall valid parameters and too numerous and dynamic for explicit definition of all parameters.

2.2.5 Probabilistic Approaches

An important category for classification in general and context inference in particular is the application of probability theory. The uncertainty inherent in measurements and propositions about situations is represented by probabilities that are encoding degrees of belief in the proposition with values between 0 and 1. The propositions can depend on each other, which is modelled as conditional probabilities. Most approaches using probabilities for classification apply *Bayesian Networks*, which represent and efficiently exploit the conditional independence of propositions. They were named after Bayes' theorem which is the key to their evaluation.

There are three (binary) discriminators which define eight general categories of probabilistic classifiers:

1. Independent or dependent propositions: A "Naïve Bayes" approach assumes mutual independence of all propositions given the target proposition. This is a simplification that is used by some approaches to save complexity in modelling, learning and evaluation.

2. Static or dynamic inference: Probabilistic approaches can include or neglect the temporal domain of a classification domain. I.e. they take into account only the current status of the propositions or also their history.

3. Exact or approximate inference: In some cases the calculation of the posterior probability is too time consuming or also too complex, so the posterior probability is only estimated given the known probabilities.

For classification, a model of the probabilities and dependencies of the relevant propositions (hence for context inference of lower and high level context) is used, classification calculates then the most probable state of the target proposition, the high level context. This model can be learnt from existing data, but also set manually by human experts as they can be interpreted easily. Also combinations of learnt and set probability models are possible.

For probabilistic inference, tools like SMILE (Structural Modeling, Inference and Learning Engine) [116], the Weka [137] framework and its extension Weka-Parallel [111] provide a large set of different approaches, all allowing for deductive and inductive reasoning, which can be used for standard classifications.

For specialised classification and inference there are to date many applications of Bayesian techniques, in particular for activity recognition. An example is Korpipää's work in [214] in which Bayesian networks are used for the determination of activities of daily life in a smart home environment, mainly based on sound sensor data.

For generic context inference there are also some applications in the literature. For instance Gu et al. use it in [124] to infer the probabilities of RDF statements; Beamon in [108] combines BNs with first order logic as a general and flexible reasoning approach in her hierarchical hybrid context reasoning engine (HyCoRE).

Probabilistic inference comes closest to the requirements presented at the beginning of this section on page 20, as it can deal with imperfection of data, can fuse several possibly contradicting input sources and its modelling of situations is expressive and can be adapted by experts.

Therefore, probabilistic Bayesian approaches have been chosen for context inference in this work. They lack however adaptation to ubiquitous computing, like distributed processing, personalisation and efficient evaluation even in large scale environments. The research at hand shall address these deficits. Section 2.3 will therefore explain the fundamentals, the concepts and the different existing Bayesian techniques in detail.

2.2.6 Further Approaches

Beyond the five main categories presented above, a few further approaches have been proposed in literature.

For instance Krause et al. in [78] use a collection of density based database techniques (e.g. k-means) and probabilistic models (e.g. Bayesian Networks and Hidden Markov Models) to learn and to identify context, however in a completely unsupervised way, which makes the data unrequestable and therefore unusable in cooperative ubiquitous computing.

Another kind is *Case Based Reasoning*, used for instance by Kofod-Petersen et al. in [128]. Case based reasoning is a form of analogical reasoning where for an inference request with no available solution an analogous problem is searched (*Retrieve-Phase*), its solution is reused (*Reuse Phase*), revised to fit the problem better (*Revise phase*) and stored for later use (*Retain Phase*) [106]. Thereby, Kofod-Petersen classifies current situations based on existing ones in the user's case base. Finding the most appropriate existing case however is difficult and the revision to the current phase is error-prone and requires strong user involvement.

Also the *Dempster Shafer Theory* [135] is used sporadically, e.g. by Wu in [139]. This mathematical theory is designed to represent the belief in a proposition rather than the probability of a proposition. It is often used for sensor fusion where the belief in a hypothesis is made up by the combination of the probabilities and beliefs in the subsets of the hypothesis [109].

Finally, some approaches use multimodal reasoning where different reasoning and inference mechanisms are incorporated for the same or different kinds of context, for instance Dargie in [114], Krause in [78] or the Persist Framework in [120].

Multimodal reasoning is desirable for the heterogeneous ubiquitous computing and shall be enabled by the concepts proposed in this thesis. The remaining approaches presented in this section do not support features which are important for ubiquitous computing and which would not be realisable with probabilistic techniques.

2.3 Bayesian Techniques

As seen in the last section, probabilistic methods and in particular Bayesian methods fit best the general requirements for classification and context inference. This section shall give a general, theoretical introduction into Bayesian techniques, ranging from terminology and semantics over learning the respective models to inference in them, both in a static and a temporal way.

2.3.1 Basics of Probability Theory

There are many debates about foundations and meanings of probabilities. In particular there are the frequentist, the objectivist and the subjectivist views. Details about them and their distinction are described by Russell and Norvig in [66]. This work adopts the subjectivist view.

Terminology:

From the subjectivist point of view, probabilities, noted as P, represent a degree of belief in *propositions*, statements about the reality. A mathematical, formal theory of them has been developed by Kolmogorov in the 1930s [68].

A basic concept is the *random experiment*. A random experiment is a process with an observable outcome, it can be natural or set up deliberately. The *sample space* Ω is the set of all possible outcomes which is assumed to be finite or countably infinite. An experiment has a random outcome if the result of the experiment cannot be predicted with absolute certainty. An *event* $\omega_i \in \Omega$ is a collection of possible outcomes of an experiment, the elements of this collection are *elementary* (also called *atomic*) events. An event is said to occur as a result of an experiment if it contains the actual outcome of that random experiment [51].

Based on random experiments, *Random Variables* (RVs) are defined.

Definition 12 *Random Variable*:
A random variable X is a function on the sample space Ω:

$$X : \Omega \to \mathbb{R} ,$$

of a random experiment with a defined probability P over Ω.

The *probability distribution* $\mathbf{P}(X)$, written in bold face, is defined as:

$$\mathbf{P}(X) = \{(x_i, P(X = x_i)|x_i \in X(\Omega)\} ,$$

where $X = x_i \Leftrightarrow \{\omega \in \Omega : X(\omega) = x_i\}$.

Analogously to $P(X = x_i)$, $P(X \in I)$ can be defined with the probability P over Ω: $X \in I \Leftrightarrow \{\omega \in \Omega : X(\omega) \in I\}$.
$X(\Omega)$ is called the *value range* of X (alternatively also called *domain* or *state space*).

Definition 12 holds for both, continuous and discrete, finite value ranges which represent finite subsets of $\mathbb{R}$. If X has a discrete value range, it is called a *discrete random variable*. A special case of a random variable with discrete value range is constituted by *boolean random variables* which have only the values *true* and *false*: $X(\Omega) = \{0, 1\}$ with $0 = false$ and $1 = true$.

In the case of discrete RVs the probability distribution can be represented as a vector of the probabilities for all values in its value range, or, more frequently as a probability table. The probability distribution of a continuous RV is represented as a *probability density function* (PDF).

With the notation, this work follows Russell and Norvig in [66]:
Random variables shall be represented with capitalised names, e.g. X, sets of RVs are denoted in bold face like $\mathbf{X}$.

Values shall always be represented with a lower case initial letter, e.g. x_i. A shorthand notation used throughout this work is the following: $P(X = x_i) = P(x_i)$. $X = x_i$ can be called an *instantiation* of X. An instantiation of the set of RVs $\mathbf{X}$ is a combination of values $X_i = x_{ij_i} \forall X_i \in \mathbf{X}$, denoted as $\mathbf{x}$.

The probability of a set of events or values of RVs is also defined. It is written equivalently in the notations $P(X = x_i \cap Y = y_j) = P(X = x_i \wedge Y = y_j) = P(X = x_i, Y = y_j)$. Analogously, the probability distribution of combinations of RVs, the so called *joint probability distribution* (JPD), is written as $\mathbf{P}(X = x_i \cap Y = y_j) = \mathbf{P}(X = x_i \wedge Y = y_j) = \mathbf{P}(X = x_i, Y = y_j)$. With this notation $\mathbf{P}(\mathbf{X})$ is the probability distribution of a set of RVs $\mathbf{X}$ and $P(\mathbf{X} = \mathbf{x}) = P(\mathbf{x})$ is the probability of the instantiation $\mathbf{x}$ of $\mathbf{X}$.

A JPD of discrete RVs is represented by a table assigning probabilities to every instantiation of the cross product of all involved value ranges. A *conditional probability distribution* (CPD) of X and Y encodes $P(X = x_i|Y = y_i) \forall i, j$ and is denoted $\mathbf{P}(X|Y)$. For discrete RVs, the CPD is represented in a *conditional probability table* (CPT).

Random variables whose values are observed are called *evidence*. It is distinguished between *soft evidence* which specifies a probability distribution for the evidence, and *hard evidence*. The latter can be seen as a probability distribution in which one value e has $P(e) = 1$ or as a direct instantiation of the evidence node E with $E = e$. Both views are equivalent. If not further specified *evidence* will refer to hard evidence.

The probability distribution of X in the absence of other information is called *prior probability distribution* or short *prior*. The probability distribution of X with given evidence E, $\mathbf{P}(X|E)$, is called the *posterior* (probability distribution).

Calculating with probabilities:

Calculating with probability is based on Kolmogorov's axioms [66]. A first fundamental rule for Bayesian techniques based on these axioms is the determination of a conditional probability with unconditional probabilities. It is defined by the *product rule* of the events x and y:

$$P(x \wedge y) = P(x|y)P(y) = P(y|x)P(x) \tag{2.1}$$

Bayes's theorem uses this rule and transforms it to calculate $P(x|y)$ based on $P(y|x)$. It can be generalised to the probability distribution of random variables as [66]:

$$\mathbf{P}(Y|X) = \frac{\mathbf{P}(X|Y)\mathbf{P}(Y)}{\mathbf{P}(X)} \tag{2.2}$$

Note that this notation does not express a division by a vector or a matrix. It is a shortcut for the matrix $\mathbf{P}(Y|X)$ with entries $P(y_j|x_i) = \frac{P(x_i|y_j)P(y_j)}{P(x_i)}, \forall i, j, P(x_i) \neq 0$.

An important technique for calculating the probability of an event or the probability distribution of a random variable is called *marginalisation* or *summing out*. It calculates the wanted probabilities by summing the probabilities given z, for all elements z of the value range of another random variable Z.

$$\mathbf{P}(X) = \sum_z \mathbf{P}(X, Z = z) = \sum_z \mathbf{P}(X, z) \tag{2.3}$$

A variant is called *conditioning*. It transforms the joint probabilities in equation (2.3) with the help of the product rule in equation (2.1) to:

$$\mathbf{P}(X) = \sum_z \mathbf{P}(X|z)P(z) \tag{2.4}$$

The *chain rule* is another application of the product rule. It holds for events just like for complete random variables, which is useful for decomposing joint probability distributions:

$$\mathbf{P}(X_1, ..., X_n) = \prod_{i=2}^{n} \mathbf{P}(X_i|X_{i-1}, ...X_1) \cdot \mathbf{P}(X_1) \tag{2.5}$$

A further useful technique depends also on the product rule. If equation (2.1) is applied to a random variable X resolved to calculate $\mathbf{P}(X|y) = \frac{\mathbf{P}(X,y)}{P(y)}$ one can easily see that the factor $1/P(y), P(y) \neq 0$ does not change for all possible values of X. It can therefore be considered a normalisation factor which eases some calculations.

The notion of *independence* helps to ease many calculations, as it reduces the size of the domain representation [66]. Two events x and y are independent, iff:

$$P(x|y) = P(x) \tag{2.6}$$

The same holds for RVs X and Y [66]:

$$\mathbf{P}(X|Y) = \mathbf{P}(X) \tag{2.7}$$

More simplifications are possible with the notion of *conditional independence*. Two RVs X and Y are conditionally independent given a third variable Z, iff:

$$\mathbf{P}(X, Y|Z) = \mathbf{P}(X|Z)\mathbf{P}(Y|Z) \tag{2.8}$$

This is equivalent with $\mathbf{P}(X|Y, Z) = \mathbf{P}(X|Z)$. One can say that Z *separates* X and Y [66].

With these concepts and rules, the "language" used in probability theory is more expressive than propositional logic [66]. It is a real superset, because all elements of propositional logic can also be represented with the language of probabilities, and in addition more degrees of belief than *true* or *false* can be assigned.

2.3.2 Bayesian Networks

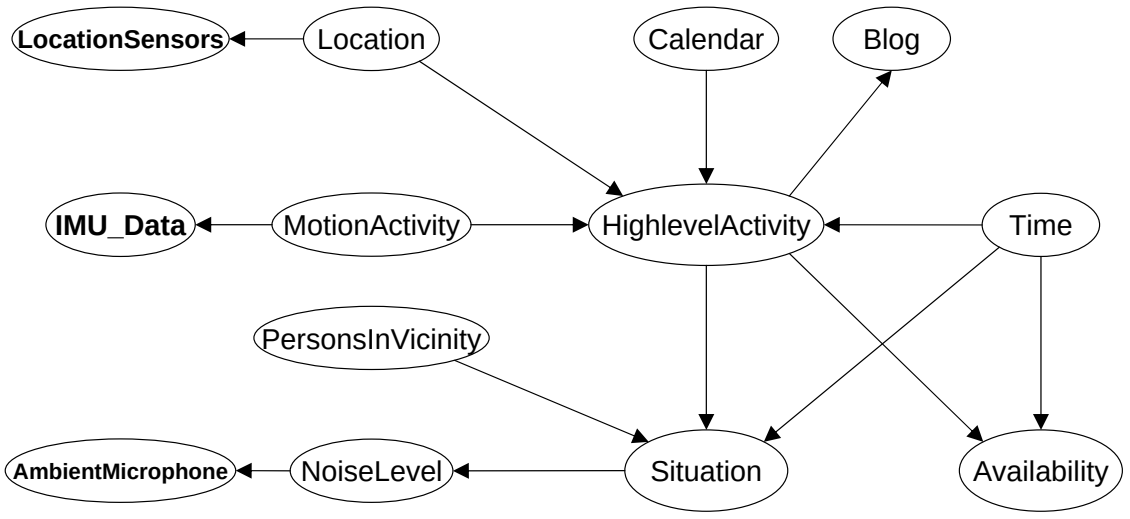


Figure 2.7: Structure of an example Bayesian Network for a grossly simplified call redirection service like the one presented in section 1.2.1. *Availability* is influenced by the *Time* of the day and the currently performed *HighlevelActivity*, which in turn is influenced by the person's *Location* and *MotionActivity*, as well as his or her *Calendar* and the current time. This high-level activity becomes manifest in a *Blog* and influences together with the $PersonsInVicinity$ the *Situation* of the environment that is reflected in the ambient *NoiseLevel* also. *LocationSensors*, IMU_Data and *AmbientMicrophone* are the sensors measuring and therefore caused by *Location*, *MotionActivity* and *NoiseLevel* respectively. As they carry evidence they are shown in boldface.

Bayesian Networks (BNs) are graphical models that represent the dependencies among random variables and give a concise specification of the full joint probability distribution. They are also known under the terms *belief network*, *probabilistic network* or *knowledge map*.

Definition 13 *Bayesian Network*:
A BN is a triplet $(\mathbf{V}, \mathbf{E}, \mathbf{P})$ with a set of RVs $\mathbf{V} = \{A_1, A_2, \ldots, A_n\}$, a cycle free set of directed dependencies $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$ between these RVs and a joint probability distribution

$$\mathbf{P}(V) = \mathbf{P}(A_1 \cap A_2 \cap \cdots \cap A_n) = \prod_{i=1}^{n} P(A_i|\mathbf{Pa}(A_i)) \ , \tag{2.9}$$

where the set $\mathbf{Pa}(A_j) = \{A_i|A_i \in \mathbf{V} \wedge (A_i, A_j) \in \mathbf{E}\}$ can be interpreted as the *parents of* A_j.

Such a graph takes advantage of the fact that with its explicit dependencies, a BN exploits the conditional independence to represent a JPD more compactly than with a full specifications of all probabilities among the events of all random variables in the network. Every RV is represented by a node or vertex in the graph, every dependency (A_i, A_j) by a directed edge from node A_i to node A_j. The set of *children of* A_j, $\mathbf{Ch}(A_j)$, of A_j is defined analogously to $\mathbf{Pa}(A_j)$.

A particular view on BNs are *Causal Networks* [63], where dependencies are interpreted as causal influence. Hence, an edge (A_i, A_j) is drawn from the cause A_i to the consequence A_j. Russell and Norvig in [66] show that this view leads to a more sparse representation of the JPD. Moreover, it makes understanding of such a network very intuitive, in particular with a graphical representation of the BN. A BN can be drawn as a *directed acyclic graph (DAG)* like the one in Figure 2.7.

With the structure (RVs and their dependencies) and the CPDs these networks contain the information known about a specific domain represented by the BN. They are a knowledge representation and maintenance format. The observation that RV $A_j = a_{j,y}$ sets $P(A_j = a_{j,y}) = 1$ and $P(A_j = a_{j,x}) = 0, \forall x \neq y$. In the case of discrete RVs, this can be interpreted as "selecting a column" of the conditional probability tables of all child nodes, and removing the node itself from the BN.

Introducing soft evidence in a RV can be represented equivalently by adding a boolean child node to the observed node [50]. The CPT of the child node has to represent the same relations as the soft observations, as represented in Figure 2.8. Hard evidence in the child node is then equivalent to soft evidence in the actually observed node.

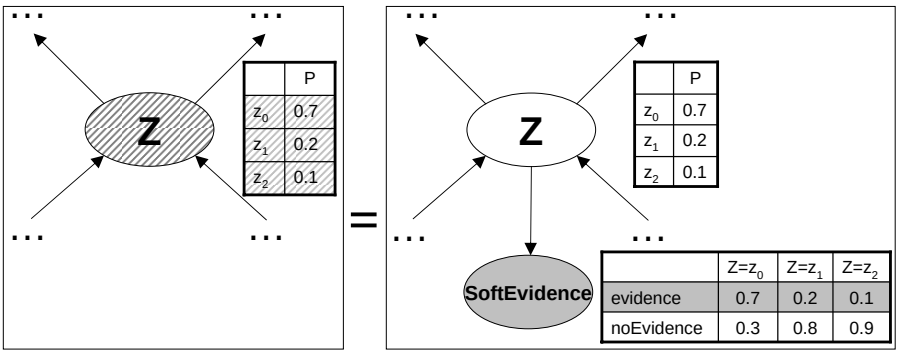


Figure 2.8: Simulation of soft evidence by hard evidence: Node Z on the left side is assigned soft evidence with a belief of 70 % in the value z_0. A table next to a node represents its probability distribution with its values in the first column; a conditional probability distribution has the values it is conditioned on in the first row. On the right side, node Z has the same belief by propagation of the belief of its special child node, called $SoftEvidence$. This node represents a binary RV with value range $\{evidence, noEvidence\}$ and the conditional probabilities that encode the degree of evidence. If the node $SoftEvidence$ is assigned hard evidence in the value *evidence*, node Z has the same (posterior) probability distribution as on the left side with soft evidence.

Note that this work focuses mainly on BNs with discrete random variables. Where necessary and possible, the value range of continuous random variables was discretised so that this assumption holds. For details how hybrid Bayesian networks can be represented using both, discrete and continuous RVs, see [66].

An important concept for BNs is *d-separation* with the "d" standing for dependence. It helps to reduce the network to relevant parts of it for given observations and a specific target RV whose state is queried. If two variables are d-separated relative to a set of

variables **Z**, then they are independent conditional on **Z** in all probability distributions of its BN.

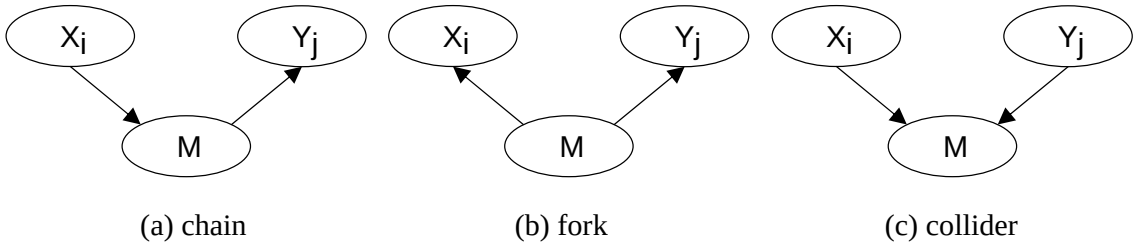


Figure 2.9: Different dependency constellations for three random variables. The d-separation of nodes depends on the direction of the arcs between them.

A set **Z** is said to d-separate node set **X** from node set **Y**, iff **Z** blocks every path from a node in **X** to a node in **Y** [63]. A *path* is a sequence of consecutive edges including at least two nodes. A path is called blocked or *d-separated* if a node on the path blocks the dependency. This is the case if the path p and the set of observed nodes **Z** are in a constellation in which

- p contains a chain $X_i \rightarrow M \rightarrow Y_j$ (see Figure 2.9 a) or a fork $X_i \leftarrow M \rightarrow Y_j$ (see Figure 2.9 b), $X_i \in \mathbf{X}$ and $Y_j \in \mathbf{Y}$, such that the middle node $M \in \mathbf{Z}$, or
- p contains an inverted fork (or collider) $X_i \rightarrow M \leftarrow X_j$ (see Figure 2.9 c), $X_i \in \mathbf{X}$ and $Y_j \in \mathbf{Y}$, such that the middle node $M \notin \mathbf{Z}$ and such that no descendant of M is in **Z**.

A construct applying d-separation is the *Markov Blanket* introduced by Pearl in [62]: Markov blanket $\mathbf{MB}(X)$ of random variable X denotes a set of nodes that d-separates X from the rest of the network $\mathbf{V} \setminus (X \cup \mathbf{MB}(X))$. The minimal Markov blanket is called *Markov Boundary*.

Assuming evidence in all nodes of the network but X, this concept implies that X is conditionally independent of all other nodes in the network, given its parents, children, and children's parents – these constitute its Markov boundary [66]. Under such conditions, the Markov boundary of a node is the only knowledge needed to infer the state of that node [62]. The values of the parents and children of a node evidently give information about that node. Its children's parents have to be included, because they can be used to explain away the node in question.

For the node *HighlevelActivity* the Markov Blanket is shown yellow in Figure 2.10 assuming that at least *Situation*, *MotionActivity* and *Location* carry evidence, so information from *LocationSensors*, *IMU_Data* and *NoiseLevel* is blocked.

A well known algorithm for determining a node's Markov boundary is the Bayes Ball algorithm developed by Shachter in [67]. Its run-time is linear in the size of the BN and visits each node only once, determining irrelevant sets and requisite information among the random variable. It simulates a ball passed from the queried node(s) to all neighbours and the ball is passed on, bounced back or blocked by the node, depending on the node's observational status and the relation to the node the ball is coming from. The exact rules are given in Figure 2.11. A node is in the Markov boundary, iff it has been visited by the ball.

Obviously the Markov boundary depends on the evidence in the BN. Soft evidence however has to be treated differently from hard evidence. The behaviour can be understood easily with the help of the equivalent construction shown in Figure 2.8, hard evidence

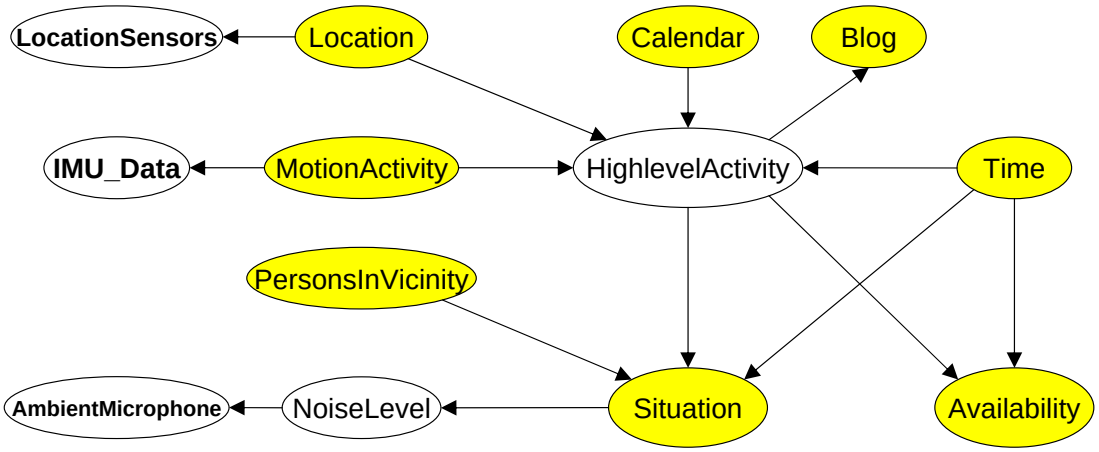


Figure 2.10: The Markov boundary (yellow) of the node *HighlevelActivity* in the example BN from Figure 2.7 when at least *Situation*, *MotionActivity* and *Location* carry evidence. It contains the node's parents, children and the parents of the children without the node itself.

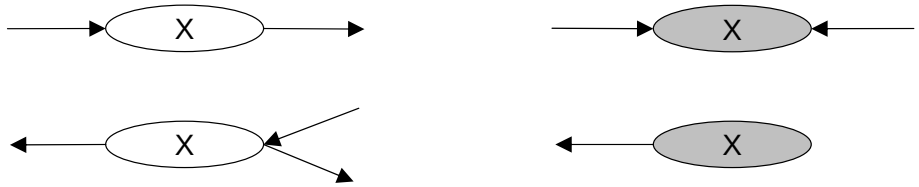


Figure 2.11: Rules of Shachter's Bayes Ball algorithm. Gray random variables carry evidence, white ones do not. The arrow to the left of the random variable determines if the ball is coming from a child or parent node, the right arrows (if any) and their direction show if the ball is passed on to parents or children.

in a child node. If the soft evidence is added to a node in the fork or chain constellation, it is not blocking the path, in the case of a collider constellation it blocks the path and therefore d-separates random variables that are connected by (only) this path.

Bayesian Networks can be grouped into *equivalence classes*, classes whose members all have equivalent network structures.

Definition 14 *Equivalent Bayesian networks*:
Two BN structures are said to be equivalent if the set of independence relationships that can be represented with one of those structures is identical to the set of independence relationships that can be represented with the other [52].

This means in particular that all networks are equivalent if and only if they have the same undirected structure and the same collider (see Figure 2.9) structures, as has been shown by Verma and Paul in [69].

There are means to simplify inference in Bayesian Networks by making additional assumptions. Such is the *Naïve Bayes* model. It says that a single cause directly influences a number of effects which in turn are all conditionally independent from each other given the cause. This results in a simple BN structure with an easy to compute JPD:

$$\mathbf{P}(Cause, Effect_1, ..., Effect_n) = \mathbf{P}(Cause) \prod_{i=1}^{n} \mathbf{P}(Effect_i|Cause) \tag{2.10}$$

Even if the assumption does not hold in reality, it yields good results in many situations at very low inference costs. For instance it is often used for spam detection in e-mail.

Other means of simplification, such as *Noisy OR*, can be found in the literature, for instance [66].

2.3.3 Probabilistic Inference

This section shall present the complexity of the inference problem in general. After that, the different approaches to probabilistic inference are introduced, ranging from exact inference over approximating the posterior probability to real-time inference, working under time constraints.

Probabilistic Inference in BNs or *belief updating* [85] means calculating $\mathbf{P}(\mathbf{X}|\mathbf{E})$ of a set $\mathbf{X}$ of RVs given evidence $\mathbf{E}$. There are many different ways for inference, all building on Bayes' theorem from equation (2.2).

Note that a further inference task is *belief revision*, also known as *Maximum A Posteriori (MAP) explanation* or *most probable explanation* (MPE). Its task is to find the most probable instantiation of some random variables, given the observed evidence. The algorithms used for belief revision are however often the same as for belief update, with only small modifications [85].

2.3.3.1 Inference Complexity

Inference complexity depends to a large extent on the structure of the DAG. Literature distinguishes (next to trivial serial structures) three structure types that are shown in Figure 2.12: BNs are called *tree structured*, when for each node in the DAG there is

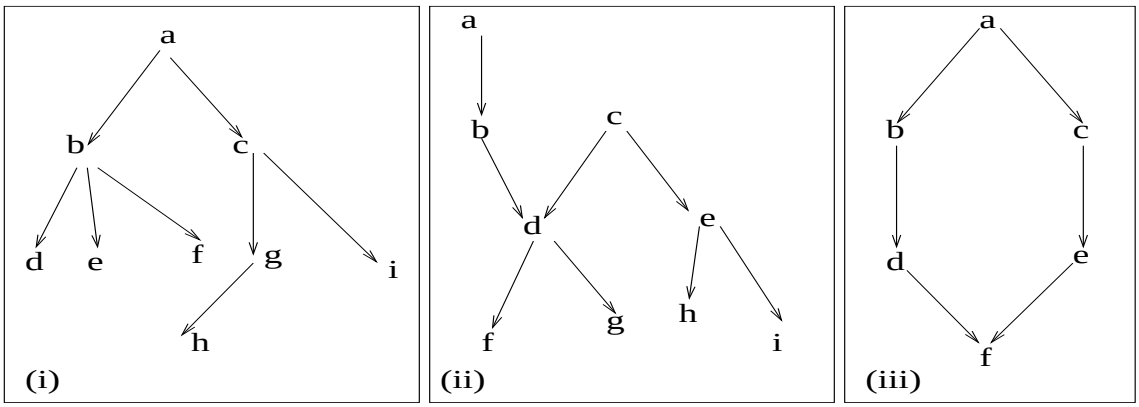


Figure 2.12: Examples of (i) a DAG of a tree structured BN, (ii) a DAG of a singly connected BN, and (iii) a DAG of a multiply connected BN taken from [96].

exactly one parent, except the root which has no parent node. For an m-ary tree and q values in the value range of each RV, $2q^2 + mq + 2q$ multiplications per update are necessary [96]. This is hence an upper bound for trees with at most m children per node and at most q values the value range of any node. As m is always smaller than the number N of RVs in the network, inference complexity is $O(N)$, if we assume an upper bound for q. As q is not directly related to N it does not have to grow with the number of nodes in the network.

In *singly connected BNs* (or also *polytrees*), where at most one undirected path exists between any two nodes in the DAG, or *multiply connected BNs*, the most general type, in which more than one undirected path exists between at least one pair of nodes, however,

belief updating is a NP-hard problem in the number of nodes N, as Cooper has shown in [82] by polynomially reducing the known NP-complete 3SAT problem [81] to probabilistic inference.

Literature often claimed that there were time efficient solutions for inference in singly connected BNs. With $O(N \cdot q^e)$ (N being the number of RVs, q the maximum cardinality of value ranges, and e the upper boundary for the number of parents of any RV) it was linear in the size of the networks, i.e. the number of entries in the CPD tables, with a bounded number of parents in the network, the complexity would also be linear in the number of nodes [89, 66]. Wu and Butz however showed in 2005 [96] that the community's opinion was inconsistent. Polynomially reducing a variant of 3SAT, 3SATV, which is also NP-complete, into a decision problem for Bayesian Networks, it can be shown that the BNs resulting from a polynomial transformation of the 3SATV problem are always singly connected. Hence also inference in singly connected BNs has to be NP-hard. Linear complexity can only be assumed if q and e are bounded. But as e depends on N (it can be up to $N-1$), then $O(N \cdot q^e)$ "can not be simply considered as linear (or polynomial)" [96].

In 1993, Dagum and Luby proved in [83] that even approximation of inference in its general case is intractable. Absolute and relative approximation could be shown to be NP-hard. Although they are not NP-complete, Dagum proves that the existence of a efficient solution would imply $NP \subseteq P$ and therefore $NP = P$, which is widely disbelieved [83].

Finally, also belief revision, in exact and approximated inference has been shown to be NP-hard, see [85] for details.

2.3.3.2 Exact Inference Approaches

Inference is simple if all nodes in the Markov boundary of the requested node carry evidence. The usage of Bayes' theorem will provide the result without significant computational burden. For the general case, many methods have been proposed, see for instance the comprehensive overview in [85]. In total, seven different concepts with many variants have been proposed which can be seen in Figure 2.13.

A first general approach is called *Variable Elimination*. It eliminates not queried variables by summing them out, taking care that factors in the probability calculation are calculated as few as possible. I.e. factors that are used several times are calculated a first time and the result is stored for further use. Marginalisation is performed only for such portions that really depend on the marginalised variable [66]. Therefore an optimal elimination ordering should be used – finding it however is NP-hard [85].

Symbolic probabilistic inference applies techniques to solve inference as a combinatorial factoring problem given a set of probabilistic distributions, the *differential* approach transforms a BN into a multivariate polynomial and computes partial derivatives for all variables. Queries are constant in time then, the construction of the model however is NP-hard [85].

Arc reversal was introduced by Shachter in [91]. Its idea is to use Bayes' Theorem and a series of other operators as often as necessary to reverse edges in the BN, until the evidence node is a direct parent of the queried node.

The *polytree* algorithm was published by Pearl in [89] and works, as the name suggests, only for singly connected networks. It is however fundamental for the class of *Conditioning* algorithms. Conditioning turns multiply connected BNs into singly connected ones by instantiation of variables that cause the *multiple* connections and applies the polytree algorithm then. The results are weighted then with the prior probability of the instantiated value. Finding the minimal set of nodes to be instantiated however is NP-hard.

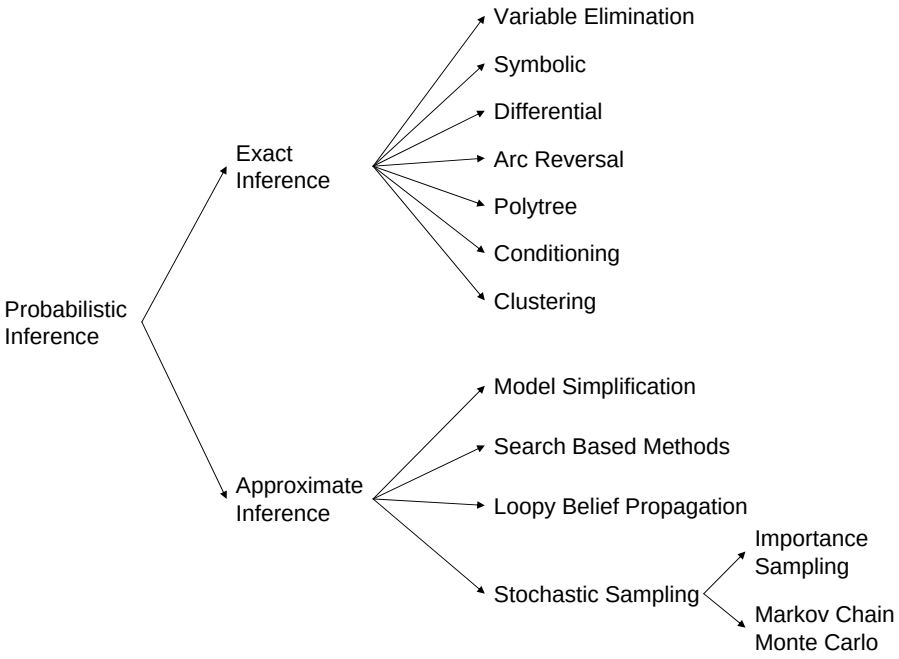


Figure 2.13: Different Approaches for Probabilistic Inference. This tree shows the categorisation of inference classes introduced by Guo and Hsu in [85].

A widely used *Clustering* algorithm was invented by Lauritzen and Spiegelhalter [88], known today under *Probability Propagation in Trees of Clusters (PPTC)*. For the linear time complexity of inference in tree-structured BNs, arbitrarily structured BNs are transformed into trees of super-nodes, the so called clusters. Inference is performed in this tree, before the queried node is marginalized from it. As this method is the most used one in general [66] and in this work, it shall be described more in detail:

Probability Propagation in Trees of Clusters:

PPTC takes advantage of proven fast inference algorithms in tree-like structured BNs. It therefore transforms the possibly multiply connected network into a tree of supernodes, so called *cliques* or *clusters* [86], in which the *message passing* algorithm can be used for propagation of probabilities.

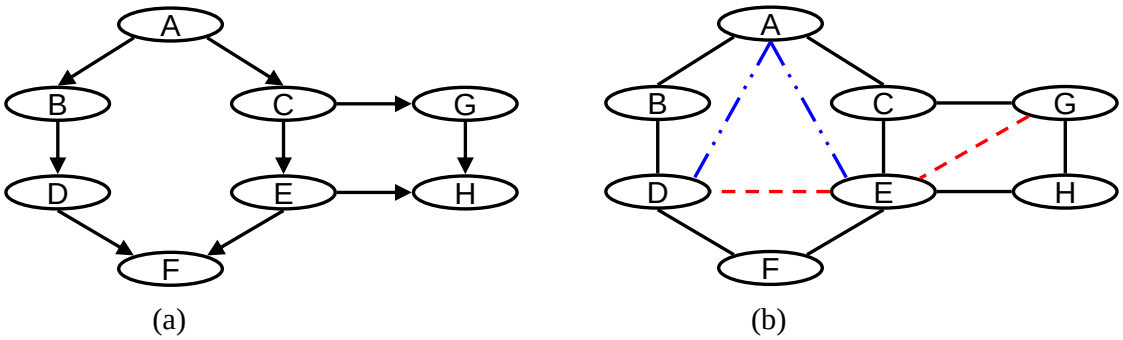


Figure 2.14: Example of a moralised and triangulated BN. The example BN on the left side is first moralised: all parents of a node are connected – represented by the red dashed lines – before the directions are removed. Then the undirected graph is triangulated, i.e. all non-adjacent nodes in cycles are connected – which is represented with the blue dashed and dotted lines.

To form these trees of clusters, also called *Join Trees*, *Junction Trees*, or short *J-Trees*, the DAG of the original BN has to be moralised (linking all parent nodes of a node, see the red dashed lines in Figure 2.14 b, and removing the directions from the edges) and triangulated.

A graph is triangulated, iff every cycle of length four or greater contains an edge that connects two non-adjacent nodes in the cycle [86], see the blue dashed/dotted lines in Figure 2.14 b) for instance. To this end, edges are added to the moralised graph until the condition is fulfilled. This process is not deterministic and depends on the arbitrary node ordering. An optimal triangulation is one that minimises the sum of the value range sizes of the clusters of the triangulated graph (and hence implicitly the number of added connections). Finding an optimal triangulation is NP-complete [80]. In practice, however, greedy, polynomial-time heuristics can be used which produce high-quality triangulations in real-world settings like the one presented in [86].

Clusters are the maximal and complete undirected subgraphs in the triangulated graph (see Figure 2.15) that represent original nodes with their related nodes. Complete means that every pair of distinct nodes is connected by an edge. Maximal means that the clique is not properly contained in a larger, complete subgraph [86]. They are assigned the potential[6] of the contained nodes. A *Sepset* (also called *separator*) is representing an edge in this supernode tree and assigned with the potential of the intersection of the two clusters linked by it.

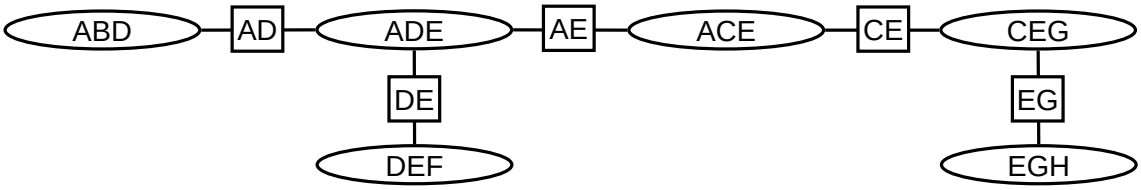


Figure 2.15: Junction Tree of the example BN of Figure 2.14 a. The triangles from Figure 2.14 b are grouped into *clusters* and connected to their neighbours via *sepsets* that carry the potential of the intersection of the adjacent clusters.

The probability propagation algorithm in such a junction tree can follow the efficient concepts of inference in tree structured BNs. The one used for PPTC is called message passing and works by passing and receiving messages between its clusters via the sepsets. Each cluster collects evidence from its neighbours only once, and once it has done that it sends out its own evidence when its neighbours request it. The end result of this is that the network gets propagated and consistent in a definitely finite number of steps [86].

To finally obtain the queried posterior probability, the probability distribution of the cluster has to be marginalised for the queried RV and normalised. Hence, once a junction tree has been set up and propagated, no complex calculations of probability are needed for inference, but only a summation over a clique.

This approach works efficiently for sparse networks, but has problems with dense networks, as its complexity is exponential in the size of the largest clique [85]. Further details on the implementation can be found in tutorial [86].

[6] A potential Φ over a set of variables $\mathbf{X}$ is defined a function that maps each instantiation $\mathbf{x}$ into a non-negative real number [88] with defined multiplication and marginalisation. A probability distribution is a special case of a potential whose elements add up to 1.

2.3.3.3 Concepts for Approximate Inference

To overcome computational hurdles imposed by the high complexity, an approach is to approximate the inference. There are in general two different classes of inference approximation Z_ϵ with boundary ϵ of $\mathbf{P}(\mathbf{X}|\mathbf{E})$ [83].

- *Relative Approximation*, if $Z_\epsilon \in \left[\frac{\mathbf{P}(\mathbf{X}|\mathbf{E}))}{(1+\epsilon)}; \mathbf{P}(\mathbf{X}|\mathbf{E})) \cdot (1+\epsilon)\right]$
- *Absolute Approximation*, if $Z_\epsilon \in [\mathbf{P}(\mathbf{X}|\mathbf{E}) + \epsilon; \mathbf{P}(\mathbf{X}|\mathbf{E}) - \epsilon]$

A large range of approximation algorithms has been invented in the literature, see [84] for a detailed overview. Figure 2.13 shows four different categories for approximated inference approaches:

Search based methods rely on the assumption that only a fraction of the joint probability space contains a majority of the probability mass. Hence they search for high probability instantiations and obtain approximations from them [85].

Loopy Belief Propagation finally is the application of Pearl's polytree algorithm [89] to multiply connected BNs. This family can perform well in situations like error-correcting codes, but can fail to converge or can give poor results in other cases [85].

Model simplification algorithms contain different approaches to simplify the network (nodes, dependencies and value ranges) until exact inference becomes computationally feasible [85].

In particular Wellman's and Liu's approach from [95] is worth mentioning. It is an approach reducing the cardinality of the random variables' value ranges to reduce inference time. Reducing the granularity of value ranges, they trade accuracy against computational efficiency. Assuming an ordered value range, they always cluster adjacent values to a *superstate*.

In order to reduce complexity moreover, the state space abstraction model ignores possibly newly introduced conditional dependencies given a superstate and conditional probabilities depending on the superstate are approximated by a uniform distribution of the merged initial states. This approach is only viable for discrete, ordered value ranges.

Stochastic simulation algorithms, the largest group for approximate inference, generate random samples or instantiations of the network according to its CPD and answer the query according to the sample set. The bigger the sample set, the more accurate, but also the more time consuming is the approximated inference.

A subgroup of stochastic simulation algorithms is called *importance sampling*. The CPDs are approximated by samples which are independent from each other. The main differences between algorithms in this group are the ways how they initialise and update the importance function and generation and weighting of the samples. In particular *forward* and *backward* sampling select the samples differently. For forward sampling, samples are created following the direction of the edges. These algorithms are not very well suited for BNs with unlikely evidence, as it takes long until they are sampled [85]. Backward sampling has advantages there, as it allows for generating samples from evidence nodes also against the direction of the edges. An improvement is the usage of an (optionally continuously updated) importance function that determines the distribution of the samples.

In *Markov Chain Monte Carlo (MCMC)* methods in contrast, samples are dependent. They perform well if there are no extreme probabilities [85] which prevent convergence. MCMC sets the evidence as specified in the query and then sets randomly values for the non-evidence nodes. Based on this initialisation it samples a randomly chosen non-evidence node by calculating its values based on the previous set values for the nodes in its Markov boundary [66]. This process is repeated iteratively.

This sampling process settles into a *dynamic equilibrium* in which the long-run fraction of time spent in each state is exactly proportional to its posterior probability and therefore returns consistent estimates for posterior probabilities. The proof is sketched in [66].

2.3.3.4 Real-Time Inference Concepts

In real-time computing systems there is a time constraint on computations. Once the deadline has passed, the utility of the result degrades massively. Real-time inference methods are designed to have an inference result within the time constraints. This is realised with two general approaches:

- *Anytime Algorithms* refine a first, imprecise calculation iteratively and therefore have always a return value for the requester.
- *Multiple Methods Approaches* use different methods for the same task in parallel.

Multiple methods approaches benefit from the fact that different approaches can be differently appropriate for the given task. With multiple solutions for a task, a trade-off can be made between solution quality and time [85]. Anytime algorithms in contrast have the advantage that they can be always interrupted and have a solution ready which is iteratively refined the longer the algorithm runs.

An example of an anytime algorithm is also Wellman's approach for "State Space Abstractions" [95]. They start with only one superstate per RV and refine by doubling the number of states.

2.3.4 Learning Bayesian Networks from Data

The last section has shown how posterior probabilities can be inferred in Bayesian networks. The latter can be constructed by human domain experts – but also learnt from previously collected evidence. In particular it is interesting that it can be reduced to another problem of inference in Bayesian networks [66].

Generally, learning of Bayesian networks can be divided into learning of the BN *parameters* (the conditional probability distributions that define the network quantitatively) and learning of the *structure*, i.e. the links between the RVs. The methods applicable to it depend on the nature of the data set which can be complete or incomplete. Structure learning for incomplete data encompasses also the most complicated case where *hidden variables* can be identified and learnt.

The following sections shall give an overview about the general problem and process of learning, parameter learning from complete and incomplete data sets as well as structure learning. The methods described here can also be applied to learning dynamic BNs (see section 2.3.5) [66], but we will focus here on methods for static Bayesian networks with discrete (and finite) random variables.

2.3.4.1 Statistical Learning

The key concepts for learning are hypotheses about the network structure, denoted by the (not directly observable) RV H and data, from the history of evidence. Data $\mathbf{d}$ are N instantiations of the observed RVs with $\mathbf{d} = \{\mathbf{d}_i\}, 1 \leq i \leq N$, where $\mathbf{d}_i$ is one instantiation of all observed random variables.

Bayesian learning calculates the probability of each hypothesis given the data and infers based on all hypotheses weighted by their probability [66]. Assuming that the hypothesis alone determines a probability distribution $\mathbf{P}(X_i)$, that the observations are

independently and identically distributed and that the observations are conditionally independent given H, $\mathbf{P}(X_i|\mathbf{d})$ depends only on the *hypothesis prior* $\mathbf{P}(h_i)$ and the *likelihood* of the data under each hypothesis $P(\mathbf{d}|h_i)$, as shown in the following equation:

$$\begin{aligned}\mathbf{P}(X_i|\mathbf{d}) = & \sum_i \mathbf{P}(X_i|\mathbf{d},h_i)P(h_i|\mathbf{d}) \\ = & \sum_i \mathbf{P}(X_i|h_i)P(h_i|\mathbf{d}) = \alpha \sum_i \mathbf{P}(X_i|h_i)P(\mathbf{d}|h_i)P(h_i) \\ = & \ \alpha \ \sum_i \mathbf{P}(X_i|h_i)P(h_i) \prod_j P(d_j|h_i) \end{aligned} \tag{2.11}$$

for a normalisation vector α.

This full Bayesian approach however is often impractical since it is computationally too expensive [75, 76], as the number of hypotheses is more than exponential in the number of nodes of the BN. This makes the full Bayesian approach intractable.

In this case approximate solutions to Bayesian learning are used. Two general approaches are *Model Selection* and *Selective Model Averaging*. The former approach rates some of all hypotheses and uses the best found, as if it were the correct model. The latter approach selects a manageable number of good models and pretends they would represent all hypothesis, i.e. be exhaustive [76].

One type of model selection uses the following approximation: it tries to select the most probable solution as the single one based on the observations that "the true hypothesis eventually dominates the Bayesian prediction" [66]. Using the hypothesis h_i with $\arg\max_{h_i} P(h_i|\mathbf{d})$ is the MAP approach. The approximation yields then

$$\mathbf{P}(X_i|\mathbf{d}) \approx \mathbf{P}(X_i|h_{\mathrm{MAP}}) \ . \tag{2.12}$$

A variant of MAP is called *minimum description length* (MDL). It attempts to minimise the size of the hypothesis and the data encoding. Applying the binary logarithm to h_{MAP} which tries to maximise $P(\mathbf{d}|h_i)P(h_i)$ one sees that this is equivalent to minimising $-\log_2 P(\mathbf{d}|h_i) - \log_2 P(h_i)$, i.e. the length of the binary representation of data and hypothesis [66].

Another commonly used simplification to MAP is the *maximum likelihood* (ML) estimator [76]. It is based on the additional assumption that the hypotheses' priors are uniformly distributed. MAP reduces hence to finding the hypothesis h_i that maximises $P(\mathbf{d}|h_i)$. This approach is a good approximation for large data sets, but has problems with small data sets [66].

2.3.4.2 Learning Bayesian Network Parameters from Complete Data

Learning network parameters $\theta_i \in [0;1]$ for some random variable X_i under the ML condition is the easiest task of learning, consisting of three basic steps [66]:

1. Description of $P(\mathbf{d}|h_\theta)$ as a function of the parameters θ_i for all i necessary to describe the BN.

2. Determination of the derivative of the *log likelihood* $L(\mathbf{d}|h_\theta) = \log P(\mathbf{d}|h_\theta)$ with respect to each parameter.

3. Determination of the maximum of $P(\mathbf{d}|h_\theta)$ by setting the derivative $\dot{L}(\mathbf{d}|h_\theta) = 0$ and resolving for each parameter.

This works quite efficiently for complete data, as the log likelihood allows for separating the learning problem for the whole BN in distinct learning problems per parameter.

However this approach has shortcomings for small data sets, where possibly not all less likely cases are observed. The full Bayesian approach to parameter learning instead uses a hypothesis prior for the parameters and updates them based on the observations. The prior probability distribution $\mathbf{P}(\Theta_i)$ thereby has to be a continuous distribution, computations with it have to be feasible efficiently and in a closed form. Candidates for the prior and posterior probability distributions are in particular beta distributions for binomial variables or the Dirichlet distribution $\mathrm{Dir}(\Theta_i, a_{i,1}, ..., a_{i,r})$ [66, 75, 71] for multinomial variables with the following density:

$$\mathrm{Dir}(\Theta_i, a_{i,1}, ..., a_{i,r}) \quad = \quad \alpha \prod_{k=1}^{r} \theta_k^{a_{i,k}-1} \ , \tag{2.13}$$

where $a_{i,k} > 0, 1 \leq k \leq r$ are called *hyperparameters*, $\Theta = (\theta_1, ...\theta_r) \in [0;1]^r$ and the normalising constant $\alpha = \frac{\Gamma(a_i)}{\prod_{k=1}^{r} \Gamma(a_{i,k})}$ and $a_i = \sum_{k=1}^{r} a_{i,k}$. Γ denotes the well known Gamma function [70] which satisfies $\forall x > 0 : \Gamma(x+1) = x\Gamma(x)$ and $\Gamma(1) = 1$. For positive integers $\Gamma(x) = (x-1)!$.

Heckerman shows in [75] that prior distributions can be calculated as follows:

$$P(X_i = x_{i,k}|\mathbf{d}) = \int \theta_{i,k} Dir(\Theta_i, a_{i,1} + N_1, ..., a_{i,r} + N_r) d\Theta_i = \frac{a_{i,k} + N_{i,k}}{a_i + N_i} \ , \tag{2.14}$$

with the number $N_{i,k}$ of observations $\mathbf{d}_n \in \mathbf{d}$ containing $X_i = x_{i,k}$ and $N_i = \sum_k N_{i,k}$.

For learning the conditional probabilities of $X_i = x_{i,k}$ the parameter Θ_i has to be extended to represent all values $\theta_{ijk} = P(x_{i,k}|\mathbf{pa}_j(X_i))$ with the instantiation j of the parents $\mathbf{Pa}(X_i)$ of X_i. Accordingly, the Dirichlet distribution depends on hyperparameters a_{ijk}:

$$\mathbf{P}(X_i = x_{i,k}|\mathbf{Pa}(X_i) = \mathbf{pa}_j(X_i), \mathbf{d}) = \mathbf{P}(X_i = x_{i,k}|\mathbf{pa}_j(X_i), \mathbf{d}) = \frac{a_{ijk} + N_{ijk}}{a_{ij} + N_{ij}} \ , \tag{2.15}$$

with the number N_{ijk} of observations of observations in $\mathbf{d}$ containing $X_i = x_{i,k}$ and the instantiation j of its parents $\mathbf{Pa}(X_i) = \mathbf{pa}_j(X_i)$, $N_{ij} = \sum_k N_{ijk}$, and $a_{ij} = \sum_k a_{ijk}$.

For implementation the hyperparameters $a_{i,k}$ and a_{ijk} are chosen as small positive real numbers in relation to the N_i and N_{ij} respectively. They must not be 0 to avoid zero probabilities which might not reflect the reality and are due to a too small data set $\mathbf{d}$.

An idea for learning is to incorporate the parameters as continuous RVs in the Bayesian network, as shown in Figure 2.16. For every evidence configuration $\mathbf{d}_n \in \mathbf{d}$ respective nodes $X_{i,n}$ can be added to a BN linking to the same parameter node $\Theta_{i|Pa(X_i)}$. Learning a parameter θ is reduced then to inference in Bayesian networks and finding the posterior probability distribution of Θ [66].

2.3.4.3 Learning Bayesian Network Parameters from Incomplete Data

A more challenging task is the learning of the BN parameters, if some (or all) cases (joint measurements) in $\mathbf{D}$ have unobserved random variables. The approach described above does not work if any $X_{i,n}$ is unobserved, as the the corresponding parameter $\Theta_{i|Pa(X_i)}$ is then not d-separated from the rest of the network and cannot be inferred individually any more. The fact that a RV value is missing may be completely random or again dependent in some way on the RV itself or other RVs in the scope.

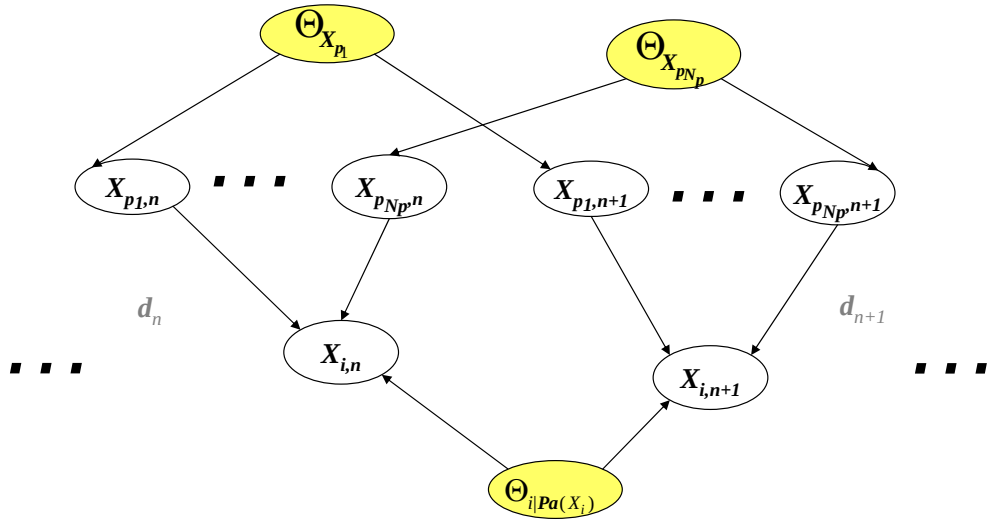


Figure 2.16: A Bayesian network that corresponds to a Bayesian learning process, incorporating the nodes $\Theta_{p_1}, ..., \Theta_{p_{Np}}$ for X_i's parent nodes $X_{p_1}, ..., X_{p_{Np}}$ and $\Theta_{i|Pa(X_i)}$ for X_i representing the parameters for their probability distributions.

For missing data algorithms can be employed from the class of *expectation maximization (EM)* introduced in [74]. EM uses probabilistic inference based on the known data to complete the data set and re-estimates the RVs parameters then including the estimated data (as if they were observed) until a local maximum is found.

If $\mathbf{x}$ are all the observed values in all evidence RVs and $\mathbf{Z}$ are all the hidden variables in all the evidence, then the EM algorithm can be formulated as [66]:

$$\theta^{(i+1)} = \arg\max_{\theta} \sum_{\mathbf{z}} P(\mathbf{Z} = \mathbf{z}|\mathbf{x}, \theta^{(i)}) \log P(\mathbf{x}, \mathbf{Z} = \mathbf{z}|\theta^{(i)}) \tag{2.16}$$

It consists of two steps:

1. In the *E-step*, the expected values of the hidden variable are computed. This step is represented in Eq. (2.16) in the summation, which is the expectation of the log likelihood $\log P(\mathbf{x}, \mathbf{Z} = \mathbf{z}|\theta^{(i)})$ of the "completed" data with respect to the posterior over the hidden variables given data.

2. In the *M-step* those parameters are selected that maximise the expectation of the log likelihood. This step is represented in Eq. (2.16) in the maximisation of the E-step with respect to the parameters.

Computation of the E-step is often computationally intractable and requires the usage of approximate approaches such as MCMC [66].

2.3.4.4 Structure Learning

In the last sections, methods were presented to learn probability distributions for given structures of Bayesian networks, i.e. given variable names and dependencies. These can be set up by human experts, but can also be learnt from the same data $\mathbf{d}$ (representing the random variables $X_i, 1 \leq i \leq n$) as the parameters. The generic solution to this was already sketched in section 2.3.4.1. This section shall briefly present the structure learning approach by Cooper and Herskovits from [71] when all nodes are observed in the data, before the process for structure learning with hidden nodes is sketched.

The aim is to maximise the posterior probability of the network structure $P(H|\mathbf{d})$ which is equal to $\alpha \cdot P(\mathbf{d}|H) \cdot P(H)$ where α is the normalising constant. Assuming equally likely network structures we can maximise the Cooper Herskovits score [71]:

$$P(\mathbf{d}|H) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(a_{ij})}{\Gamma(a_{ij} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(a_{ijk} + N_{ijk})}{\Gamma(a_{ijk})} \ , \tag{2.17}$$

where q_i is the number of parent configurations of node X_i (i.e. the product of the number of values in the range of all parent nodes) and r_i is the number of values in the range of X_i. The data set is used to count the numbers N_{ijk} for all $1 \leq i \leq n$, $1 \leq j \leq q_i$ and $1 \leq k \leq r_i$ which denote the number of times that joint measurement of random variable X_i with value index k and its parent configuration j was observed in the data $\mathbf{d}$. Similarly, $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. The values of a_{ijk} and $a_{ij} = \sum_{k=1}^{r_i} a_{ijk}$ are the parameters of the Dirichlet prior distribution on the conditional probabilities for node X_i. There are various fundamental approaches to assigning the priors in a practical manner [75].

A network with n nodes has about $3^{\frac{n^2-n}{2}}$ possible network structures (directed cycles are not excluded by this number). I.e. when $n = 10$, there are more than $2 \cdot 10^{21}$ possible structures and the number is growing faster than exponentially. Since these numbers are far too large for realistic n to assess with Eq. (2.17), suboptimal approaches have to be resorted to, i.e. model selection or selective model averaging as shown in section 2.3.4.1.

An often used heuristic algorithm for finding an optimal network structure (hence model selection) is the *K2* approach presented by Cooper and Herskovits in [71]. It uses greedy hill climbing to search the network space employing random restarts to avoid getting stuck in local maxima. The modularity of Eq.(2.17) is used to shorten calculations: A typical greedy hill climbing search will make one arc change (addition, removal or reversal) per step and as a result not all product terms need to be recomputed. To avoid recalculation of terms they can be cached node by node for all explored combinations of their parent nodes.

The appropriateness of every network structure is evaluated then by a score function and only the higher scoring structure is followed. The value of $P(\mathbf{d}|H)$ indicates how well the network hypothesis H encodes the dependencies between the RVs; often the best network will be many orders of magnitude better in score than the second best [75]. To reduce computational burden, Cooper and Herskovits in [71] use the log likelihood of $P(\mathbf{d}|H)$ which transforms the multiplications into additions which are less costly.

Learning of structures with missing data contains in particular the case of hidden variables where there are some RVs which are never observed. When trying to learn hidden variables, *structural EM* introduced in [74] or combinations of data imputation and EM such as described in [79] can be applied. Hidden variables can be learnt automatically by identifying areas in the network that could benefit from a new hidden variable [73]; recent work has developed algorithms with improved convergence using the information bottleneck principle [72]. To date however there is no approach efficient enough to be practical in most real-world applications [71].

Again, the main idea of structural EM is to use a current model to evaluate new structures. Iteratively the current model helps to find either better scoring parameters or a better scoring structure.

2.3.5 Dynamic Bayesian Networks

A *dynamic Bayesian network (DBN)* is a Bayesian network that represents sequences of variables. These sequences are often time-series [61], for example in speech recognition. Such sequences can be considered as a series of *time slices*, each containing random variables and their dependencies. For convenience it is assumed that the same subset of variables is observable in each time slice, although this is not strictly necessary [66]. The observable RVs are called *evidence variables* and denoted as E_i, the unobserved ones are called *state variables* and denoted X_i. Sets of variables are denoted in bold, i.e. for instance the set of evidence variables is $\mathbf{E}$, the state variables $\mathbf{X}$.

To indicate the current snapshot of a random variable, the number of the time slice is added as an index. X at time t for instance is denoted X_t, if the random variable already has an index, the time index is added after a comma. Hence the RV X_i at time t is $X_{i,t}$. A sequence of a variables X from time slice a to b is denoted $\mathbf{X}_{a:b} = \{X_a, X_{a+1}, ..., X_b\}$.

To allow handling of DBNs, three assumptions are made [66]

1. Changes are caused by a *stationary process*, i.e. the possible transitions do not change over time.

2. The current state depends on a finite history of previous states only. This is called the *Markov assumption*.

3. Observations in evidence random variables depend only on the current state.

The first two assumptions define a class of processes called *Markov chains* or *Markov processes*. The maximal length of the state history incorporated is defining the order of the Markov process. In a Markov process of order n the following equation holds:

$$\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-n:t-1}) \tag{2.18}$$

Markov processes of higher order however can be equivalently represented by first order Markov processes, by simply enlarging the time slice by the random variables from previous time slices [66]. Therefore we can reduce Eq. (2.18) to the *transition model*:

$$\mathbf{P}(\mathbf{X}_t|\mathbf{X}_{0:t-1}) = \mathbf{P}(\mathbf{X}_t|\mathbf{X}_{t-1}) \tag{2.19}$$

With the third assumption then the *observation model* or *sensor model* can be defined:

$$\mathbf{P}(\mathbf{E}_t|\mathbf{X}_{0:t}, \mathbf{E}_{1:t-1}) = \mathbf{P}(\mathbf{E}_t|\mathbf{X}_t) \tag{2.20}$$

Next to transition and observation model, a third parameter defines a DBN: the prior probability $\mathbf{P}(\mathbf{X}_0)$ for the state variables at time 0. Like that it defines the full joint probability distribution of a DBN:

$$\mathbf{P}(\mathbf{X}_{0:t}, \mathbf{E}_{1:t}) = \mathbf{P}(\mathbf{X}_0) \prod_{i=1}^{t} \mathbf{P}(\mathbf{X}_i|\mathbf{X}_{i-1})\mathbf{P}(\mathbf{E}_i|\mathbf{X}_i) \tag{2.21}$$

An example of a DBN is shown in Figure 2.17 with three state variables and two evidence variables.

A *Hidden Markov Model (HMM)* can be considered as the simplest dynamic Bayesian network. In contrast to a DBN it allows *only one state variable* while DBNs do not have any structural limitations. In a regular Markov model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a

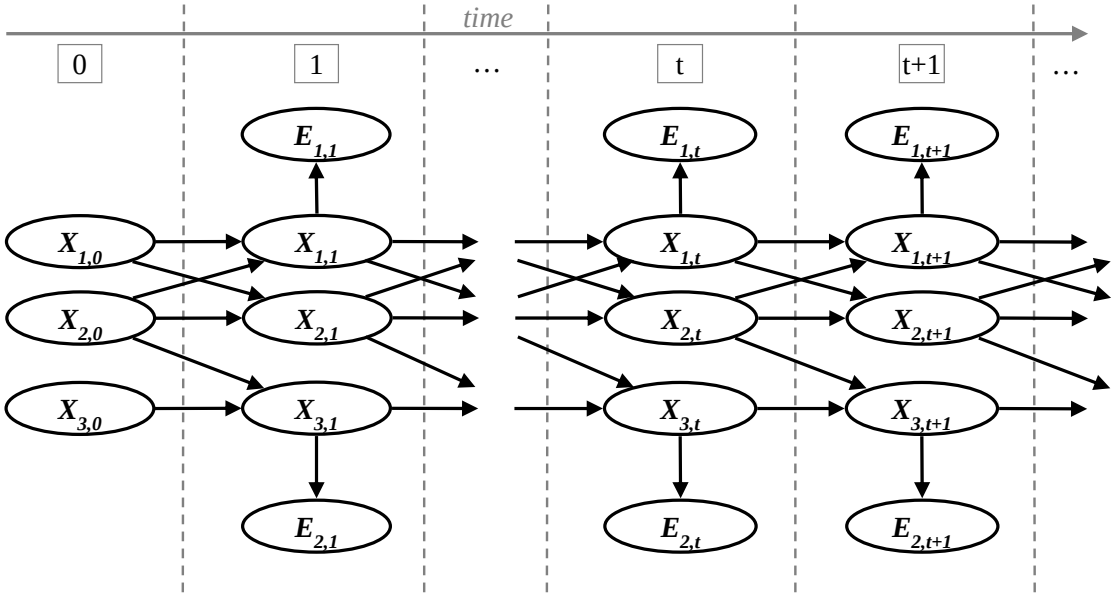


Figure 2.17: An example dynamic Bayesian network with three state variables X_1, X_2, X_3 and two evidence variables E_1, E_2.

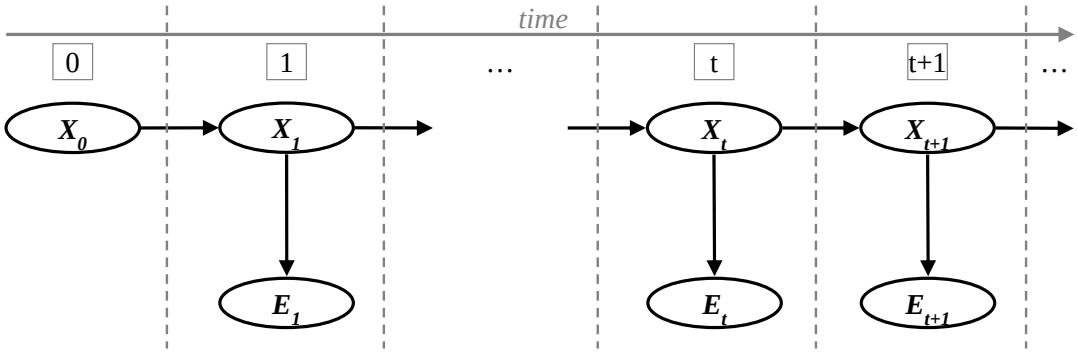


Figure 2.18: Example Hidden Markov Model with only one evidence random variable E.

hidden Markov model, the state is not directly visible, but observations dependent on the state are available [65], the evidence variables. The most simple version of a HMM is given in Figure 2.18. It is a first order HMM with only one evidence node E.

In the case when the HMM λ consists of discrete random variables it can be represented with matrices as:

$$\lambda \sim (\mathbf{A}, \mathbf{B}, \pi) \tag{2.22}$$

- The state transition matrix $\mathbf{A}$ describing the transition model: $\mathbf{A} = \{a_{ij}\}$ where $a_{ij} = P(x_{j,t+1}|x_{i,t})$, $1 \leq i, j \leq N$, and the value range of X is $x_1, ..., x_N$.
- The observation matrix $\mathbf{B}$ representing the observation model in state j, $\mathbf{B} = \{b_{jk}\}$, where $b_{jk} = P(e_{k,t}|x_{j,t})$ $1 \leq j \leq N, 1 \leq k \leq M$.
- The initial state distribution $\pi = \{\pi_i\}$ where $\pi_i = P(x_{i,0})$, $1 \leq i \leq N$ is the prior probability.

Every DBN can be transformed into a HMM, by grouping all state variables of the DBN into one supernode in a HMM and joining all value ranges, i.e. building the cross product. This means that HMM and DBNs are structurally equivalent. DBNs however,

using the properties of Bayesian networks, can represent the probability distribution of the state variables more efficiently. This is shown in [66] in an impressive example: if 20 boolean state variables were joint to a super state variable, its CPT would contain 2^{20} single probabilities. Assuming that these state variables in the DBN have all three parents, the number of probabilities to be specified is only $20 \times 2^4 = 320$.

2.3.6 Probabilistic Reasoning over Time

Reasoning in DBNs can provide different insight. The classical task is the estimation of the current step given evidence from the beginning until the current state. It is classification or context inference taking into account the history of the state, hence the calculation of $\mathbf{P}(\mathbf{X}_t|\mathbf{E}_{1:t})$. This process is called *filtering*.

The easiest approach to calculate this, is to use the standard approaches described in section 2.3.3, as DBNs are a subclass of Bayesian networks. To do so, one has to construct the full BN repesentation of the DBN by replicating the time slices from 0 to t. This technique is called *unrolling* [66]. A naïve application of it however is dependent on all variables $\mathbf{E}_{1:t}$ – and therefore linearly increasing with t, which becomes intractable very soon.

Besides filtering, one could also *predict* future states, *smooth*, i.e. calculate a past state based on all available evidence up to the current state, or find the MPE for a sequence of given evidence. This all can be reached with methods very similar to filtering, which is why the remainder of this section focusses only on filtering.

Filters or *Recursive Bayesian Estimation Techniques*, also known as the *Forward algorithm* [66, 65], represent a less naïve class of inference algorithms, using the assumptions presented in section 2.3.5. Inference of a set of state variables $\mathbf{X}$ at time $t+1$ given hard evidence $\mathbf{E} = \mathbf{e}$ from time 1 to $t+1$ is calculated as follows:

$$
\begin{aligned}
\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) &= \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}, \mathbf{e}_{t+1}) \qquad (2.23)\\
&= \frac{\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}, \mathbf{e}_{1:t}) \cdot \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})}{P(\mathbf{e}_{t+1}|\mathbf{e}_{1:t})} \text{ (by Bayes' theorem in Eq. (2.2))}\\
&= \alpha\ \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}, \mathbf{e}_{1:t}) \cdot \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t})\\
&= \alpha\ \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \cdot \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) \text{ (by Sensor Model in Eq. (2.20))}\\
&= \alpha\ \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}, \mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) \text{ (Conditioning, Eq. (2.4))}\\
&= \alpha\ \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) \text{ (Markov Property)}
\end{aligned}
$$

$\alpha = \frac{1}{P(\mathbf{e}_{t+1}|\mathbf{e}_{1:t})}$ is a normalising factor. As can be seen, the calculation depends on two stages:

- Prediction stage: The transition model is used to predict the state probability distribution function from one measurement to the next.

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) = \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) \qquad (2.24)$$

- Update stage: The latest measurement is used to modify the prediction PDF with the observation model.

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) = \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \cdot \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) \qquad (2.25)$$

With this recursive approach, constant time and space requirements per update step can be achieved [66]. It represents hence an improvement over naïve unrolling, by making posterior probability updates from time t to $t+1$ independent of the complete history, whereas the resulting posterior still incorporates the information of the complete history. This is essentially realised by summing out the state variables of the previous time step – hence it is a form of *variable elimination* presented above.

Analysing the formulae, Khider observes in [58] the following:

- The prediction uses the transition model and the given previous PDF $\mathbf{P}(X_t|\mathbf{e}_{1:t})$ at time t to expect a PDF for the current state.
- At time step $t+1$, a measurement $\mathbf{e}_{t+1}$ becomes available and is used to update the prediction. The arrived measurement gives an indication regarding the correctness of our prediction.
- A closer look at the update equation shows that the update equation is the prediction equation multiplied by the likelihood $P(\mathbf{e}_{t+1}|X_{t+1})$ and a constant. The likelihood is calculated from the observation model. So update is the prediction weighted with the received measurement at time $t+1$.
- In the Bayesian process the prediction and the update are repeated as time evolves. So prediction keeps going until a measurement is received. When a measurement is received an update happens and then prediction continues.

Also for Bayesian Filters, exact and approximated estimation approaches have to be distinguished, i.e. optimal and sub-optimal estimators for discrete and continuous value ranges. Important filters are presented in the following subsections.

2.3.6.1 Grid Based Filter

A *Grid Based Filter* provides an optimal recursion of the filtered probability distribution $\mathbf{P}(\mathbf{X}_t|\mathbf{E}_{1:t})$, if the state space is discrete and consists of a finite number of states. It relies on discrete, piecewise constant representations of the probability distributions [53].

It can apply directly the probability distributions of equations (2.24) and (2.25) and finds the most likely value of the target node X given evidence $\mathbf{e}$ by calculating:

$$\arg\max_{x_{i,t}} P(x_{i,t}|\mathbf{e}_t) \tag{2.26}$$

A key advantage of the grid based filter is that it can cope with arbitrary distributions over the discrete state space. The disadvantage of grid-based approaches is their computational complexity, which makes them applicable to low dimensional estimation problems only, such as estimating one of seven motion related activities of a person (cf. section 5.3).

If the underlying model is a Hidden Markov Model, there are efficient ways to calculate the posterior based on matrix operations. The transition and observation model can then be represented as matrices A and B as shown in Eq. (2.22).

With these matrices, the posterior can be calculated as a series of matrix operations and Eq. (2.25) can be represented as:

$$\mathbf{P}(X_{t+1}|\mathbf{e}_{1:t+1}) = \alpha B A^T \mathbf{P}(X_t|\mathbf{e}_{1:t}) \tag{2.27}$$

Grid based methods can also be used for approximate filtering in cases where the state space is continuous. Approximate grid based methods therefore quantise the state space into n cells in order to be able to approximate the posterior density with a standard Grid based approach.

2.3.6.2 Kalman Filter

The Kalman filter [57] is the most widely used variant of optimal Bayesian Filters, named after its inventor Rudolf E. Kalman. It postulates two conditions: The transition model and the observation model are *linear* and all state and evidence variables are *normally distributed*. Despite these strong assumptions, Kalman filters have been applied with great success to various estimation problems [66].

The Kalman filter calculates the probability distribution over X by unimodal Gaussian distributions, represented by their mean and variance, see Figure 2.19. While the mean gives the expected state the variance represents the uncertainty in the estimate in the one dimensional case.

In the multivariate case a Gaussian is represented by a mean vector μ and a covariance matrix Σ: $\mathbf{P}(\mathbf{X}_0) \sim N(\mu_0, \Sigma_0)$. Gaussians remain closed under Bayesian prediction and update [66], i.e. also the results of a filter step is a Gaussian.

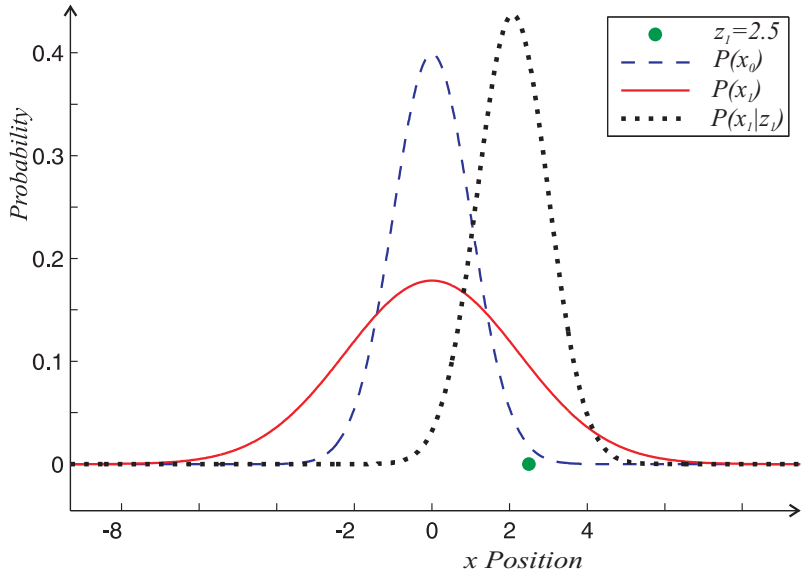


Figure 2.19: Example for Kalman Filtering from [66]. A prior given by $\mu_0 = 0$ and $\sigma_0 = 1$, transition noise $\sigma_x = 2$, sensor noise $\sigma_z = 1$ and a first observation $z_1 = 2.5$. $\mathbf{P}(X_1)$ is predicted from $\mathbf{P}(X_0)$ and updated then to $\mathbf{P}(X_1|z_1) \sim N(2.08, 0.83)$.

The big advantage of this property of Gaussians is that the representation of the probability distribution hence stays constant, while in general "filtering with continuous or hybrid (discrete and continuous) networks generates state distributions whose representation grows without bound over time" [66].

Transition and sensor models can be transformed due to the linearity of the model linearly with additive Gaussian noise to:

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{X}_t) \sim N(\mathbf{F}\mathbf{X}_t, \mathbf{\Sigma}_x) \tag{2.28}$$

$$\mathbf{P}(\mathbf{Z}_t|\mathbf{X}_t) \sim N(\mathbf{H}\mathbf{X}_t, \mathbf{\Sigma}_z) \tag{2.29}$$

$\mathbf{F}$ and $\mathbf{\Sigma}_x$ are matrices describing the transition model and transition noise covariance, $\mathbf{H}$ and $\mathbf{\Sigma}_z$ for the observation model analogously. The outcome of prediction and update can be shown to be [66]:

$$\mu_{t+1} = \mathbf{F}\mu_t + \mathbf{K}_{t+1}(\mathbf{Z}_{t+1} - \mathbf{H}\mathbf{F}\mu_t) \tag{2.30}$$

$$\mathbf{\Sigma}_{t+1} = (\mathbf{I} - \mathbf{K}_{t+1}\mathbf{H})(\mathbf{F}\mathbf{\Sigma}_t\mathbf{F}^T + \mathbf{\Sigma}_x)\ , \tag{2.31}$$

where $\mathbf{I}$ is the identity matrix and the *Kalman gain matrix*, $\mathbf{K}_{t+1}$, is defined by:

$$\mathbf{K}_{t+1} = (\mathbf{F}\boldsymbol{\Sigma}_t\mathbf{F}^T + \boldsymbol{\Sigma}_x)\mathbf{H}^T(\mathbf{H}(\mathbf{F}\boldsymbol{\Sigma}_t\mathbf{F}^T + \boldsymbol{\Sigma}_x)\mathbf{H}^T + \boldsymbol{\Sigma}_z)^{-1} \tag{2.32}$$

An *Extended Kalman Filter* (EKF) generalises the standard Kalman filter by linearisation so that it can be applied for non-linear systems also. The Gaussianity condition of the system's noise however remains as a restriction for the EKF.

To overcome the non-linearities, an EKF continually updates a linearisation around the previous state estimate, starting with an initial guess. In other words, it only considers a linear Taylor approximation [48] of the system function at the previous state estimate and that of the observation function at the corresponding predicted position.

This approach gives a simple and efficient algorithm to handle a non-linear model. However, convergence to a reasonable estimate may not be obtained if the initial guess is poor or if the disturbances are so large that the linearisation is inadequate to describe the system [58].

2.3.6.3 Particle Filter

If the DBN is not restricted in any case (i.e. transition and observation models need not be linear, the variables need not be normal distributed and the random variables are not all discrete, for example), an approximate solution must be found to solve the prediction equation. Generalising Eq. (2.24) to continuous variables yields:

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t}) = \int \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{e}_{1:t})d\mathbf{x}_t \tag{2.33}$$

As the integral cannot be numerically solved in the general case, the *Particle Filter* [49] approach uses a MCMC method as explained in section 2.3.3. It can be shown that the solution of a particle filter coincides with the exact solution as the number of particles $N_S \to \infty$ [66].

It represents probability densities over the states from $\mathbf{X}_t$ by a set of random samples, called *particles*, distributed according to the probability distribution. This random sampling is called *Monte Carlo sampling*. The continuous probability distribution is represented by N_S particles. Each of these particles has an associated weight as shown in Figure 2.20.

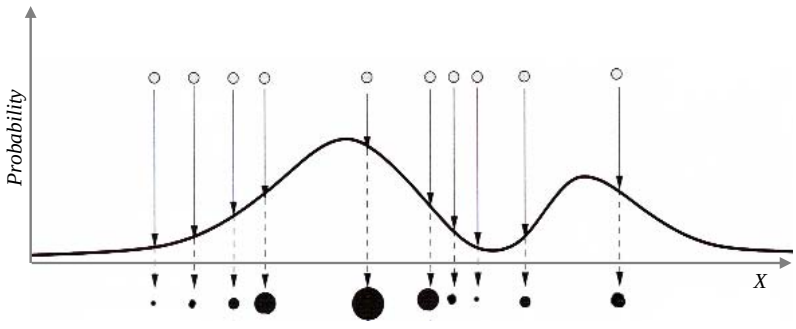


Figure 2.20: Representation of a continuous probability density function with particles from [58]. Samples (i.e. particles) in areas with a higher probability are given higher weight, which is represented by the size of the dots below the x axis.

The posterior probability distribution can then be estimated by:

$$\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{e}_{1:t+1}) \approx \sum_{i=1}^{N_S} w_{t+1}^i \delta(\mathbf{X}_{t+1} - \mathbf{x}_{i,t+1}) \ , \tag{2.34}$$

with the number of particles N_S, the sampled states $\mathbf{x}_{i,t+1}$ at time $t+1$ and the weights w^i_{t+1} of the sampled states, where $1 \leq i \leq N_S$ [49]. The weights can be defined by:

$$w^i_{t+1} = \alpha w^i_t \frac{P(\mathbf{e}_{t+1}|\mathbf{x}_{i,t+1})P(\mathbf{x}_{i,t+1}|\mathbf{x}_{i,t})}{q(\mathbf{x}_{i,t+1}|\mathbf{x}_{i,t}, \mathbf{e}_{1:t+1})} \ , \tag{2.35}$$

where α is a normalisation constant and q is the proposal function, the so called *Importance Density* [49].

There are different approaches to particle filters. The general particle filtering is realised in a *Sequential Importance Sampling (SIS)* algorithm which is extended for resampling (i.e. creating a new population of samples after an estimation) by *Sampling Importance Resampling (SIR)* filters, *Auxiliary SIR (ASIR)* filters or *Regularised (RPF)* particle filters [49]. In all these approaches the transition model is used as importance density to draw samples representing an approximation of the posterior, which is then identical to the prediction step, and the likelihood is used to weight the particles (update stage). Inverse to these methods, a *Likelihood Particle Filter (LPF)* uses an importance density based on the likelihood and the prior for weighting the particles. In the case when the likelihood is much tighter than the prior (more accurate than the prior), and since the importance density is an approximation of the prior, then using the likelihood for the importance density is expected to improve performance [49].

The implementation of a SIR particle filter follows the following steps [66]:

1. A population of N_S particles is created by sampling from the prior $\mathbf{P}(\mathbf{X}_0)$.
2. Prediction is calculated for each particle using the transition model $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)$.
3. Each particle is weighted by the likelihood it assigns to the new evidence $P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1})$.
4. The population is resampled to generate a new population of N_S particles. Each new particle is selected from the current population. The probability that a particular particle is selected is proportional to its weight.
5. Prediction is started over again with the new population at step (2).

The computational burden of particle filters depends on the number of particles chosen and processed. From experience [190], it is computationally more demanding than Kalman filters to reach comparable solution quality. On the other hand its big advantage is the general applicability to all probability distributions, transition and noise models.

2.4 Summary

This section has provided the fundamentals for this research. It has given the definitions for the terminology used in the remainder of this thesis and explained the decision for Bayesian techniques as basis for context inference being capable to handle uncertain and missing data and having a human understandable format.

Bayesian techniques provide a big variety of useful methods for context aware systems. Learning of Bayesian Networks provides the tool kit for generating inference rules, static and dynamic Bayesian networks and their subforms allow for different kinds of inference optimised for different situation models.

Their big disadvantage is the inference complexity. Being NP-hard in the general case, there are only inference algorithms with exponential complexity. This poses a significant problem for context inference based on the available context information, especially on resource constrained mobile devices.

Chapter 3

Usage Analysis of Context Aware Mass Market Services

Since one major problem of today's context aware frameworks which this thesis shall address is scalability, a first necessary step is to illuminate the target user domain with its need and the resulting requirements for context inference.

To this end, in section 3.1 the user base for context aware services and the number of services that are going to be used in the future will be quantified. In order to fully understand context aware services, it is also necessary to consider the behaviour of context. Section 3.2 will shed some light into the dynamics of different contextual aspects.

The insight gained in these sections will be used to extract requirements for context inference in section 3.3.

3.1 Quantifying Context Usage

The objective of this section is to analyse how much context is going to be used in ubiquitous computing in the future, once it will have superseded the current paradigm of mobile computing according to the vision of Satyanarayanan [37]. To give a realistic assumption of future context usage, our assumptions are based on current usage of mobile communication and internet. The rough timeframe for this prediction shall be the next 15 to 20 years – a time horizon that seems realistic for a market penetration of ubiquitous computing.

3.1.1 Smartphones and Mobile Networks

By the end of 2010, roughly 5.3 billion users were subscribed to mobile cellular telecommunications according to the analysis of global Information and Communication Technology (ICT) trends by the International Telecommunication Union (ITU) [24], which represents today about 76 % of the world population. With roughly the same or only slightly lower growth rates as today, this number would reach 8 billion subscribers in the next fifteen to twenty years [44].

Given that access to mobile networks is nowadays already available to 90 % of the world population and 80 % of the population living in rural areas [24], a modest increase to 95 % in the next 15 years can be assumed. The growth rates of mobile cellular subscriptions were higher than 8 % during the last six years [22] supported in particular by huge growth in developing countries.

Also the number of mobile broadband internet subscriptions is increasing continuously, from 0.1 % in 2003 to globally 13.6 % by the end of 2010 [22], amounting now to just below

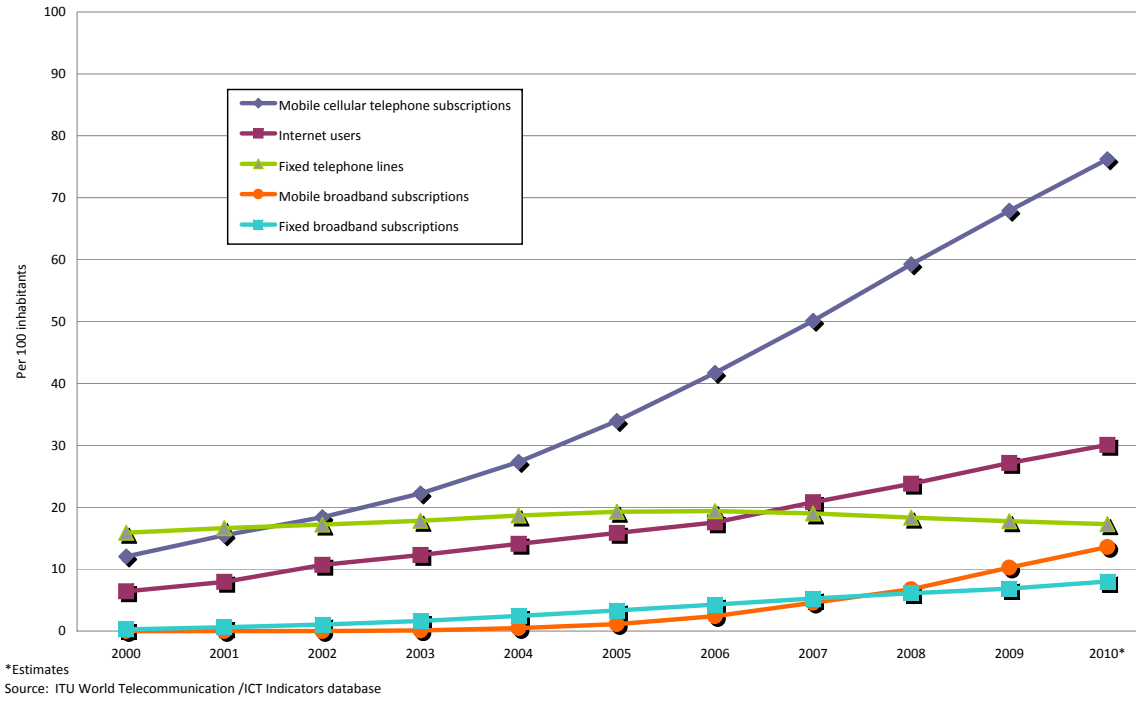


Figure 3.1: Global ICT developments, 2000-2010* [22].

one billion mobile broadband subscribers [24]. In the developed world, already more than 50 % of all mobile telecommunication users have a mobile broadband subscription [23].

As the number of mobile broadband subscriptions has already overtaken the number of fixed broadband subscriptions, its growth will probably rather follow the trend of mobile telephony than of internet usage in general.

Having reached a status slightly higher than mobile telephony in 2000, this work assumes that more than 90 % of the world population will have mobile broadband internet access in the timeframe we set above, i.e. between 2025 and 2030.

Smartphones and always available data networks are the key factors for ubiquitous computing, because with their processing power and capabilities for sensing, transmitting and receiving information they allow for *ubiquitous software services*, *ubiquitous information*, as well as *context-aware computing and technology* which were realised as key factors for Ubiquitous Computing by Abowd and Schilit in [1].

Taking into account the global population size of about 8 billion by then and that ubiquitous computing will follow mobile computing when the precondition of network access is given, it can be concluded that the global system for ubiquitous computing has to be capable to support about 7 billion users.

All these users will use mobile, context aware services on their mobile smartphones, as well as in other everyday environments like their households, their cars or workplaces. Nowadays, all of these environments are already increasingly equipped with sensors and therefore able to take into account the current context for specific applications.

Smartphones and laptops today incorporate about 10 different information sources (sensors, antennas, etc.) for temperature, accelerations, turn rates, sound, direction, position via GPS, RFID, Bluetooth, Wi-Fi and other antennas allowing for fingerprinting, as well as light and face-proximity via cameras. All the acquired information describe the context of the device's user and are used today already in many services, like the "apps"

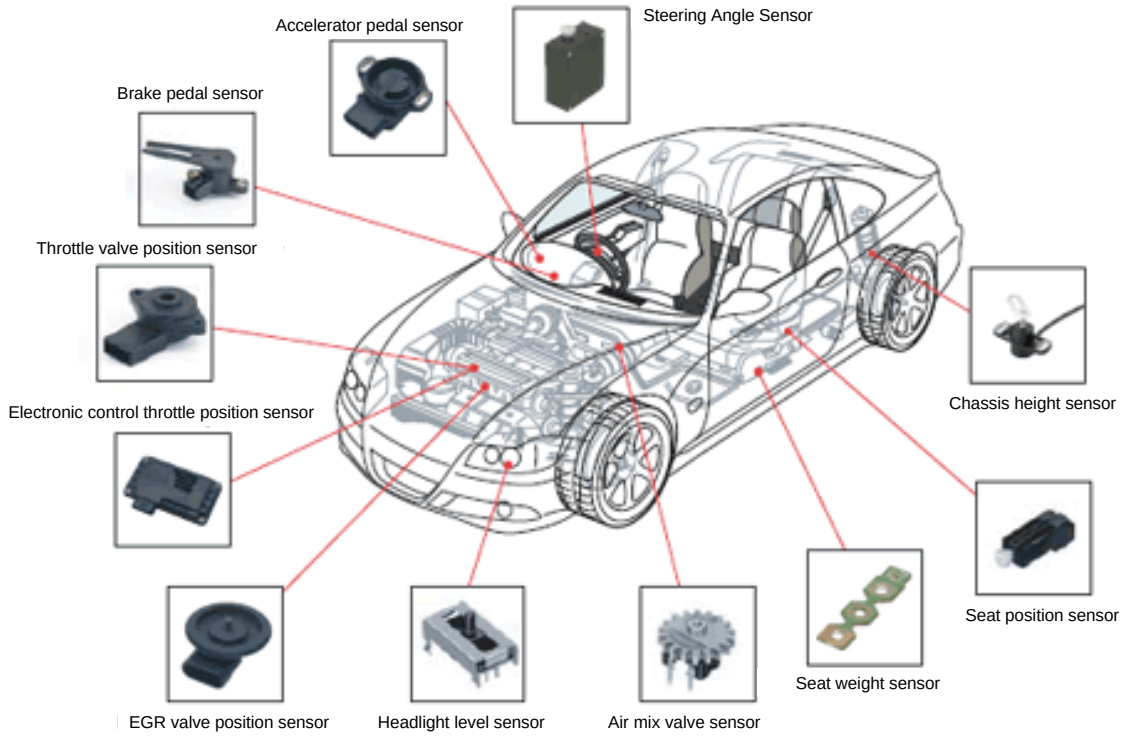


Figure 3.2: Examples for sensors in a car, available at ALPS Electric Co., Ltd. [2].

for mobile operating systems such as Apple's iOS. These apps can serve as an example how context aware services would be used in ubiquitous computing.

In the year 2010, 160 million users worldwide downloaded about seven billion iOS apps alone [4], which means about 44 apps per user. From this large number, it is still realistic to assume that four to five apps are used on a daily basis and continuously, and others only from time to time, which is also in agreement with the numbers of services used in the scenarios above, in particular the "Context Aware Day out with Friends" scenario in section 1.2.2 on page 6.

3.1.2 Vehicles

The second important environment for context aware services which must not be neglected are the currently existing 1.5 billion cars [21]. As can be seen in the scenario from section 1.2.2 on page 6, cars can use their own sensors and information from remote vehicles to control their cruise, to avoid collisions, to safe energy, but also for instance to provide information and multimedia services to their passengers. With more and more widely used services like remote heat control, unlocking via a mobile phone and similar, the car is becoming an extension of a user's smart space.

Already today's over 200 sensors (see the examples in Figure 3.2), about 50 microprocessors [29], and different wireless network interfaces (e.g. for car-to-car communication, for opening or for emergency calls) give them access to a broad range of information regarding their current context. While some of these services (e.g. CACC) are only active while the car is in motion, others have to be continuously enabled, for instance like a remote opening service that unlocks the car only if the entitled person is close to it, a car alarm, or air conditioning which starts when the driver heads towards his car.

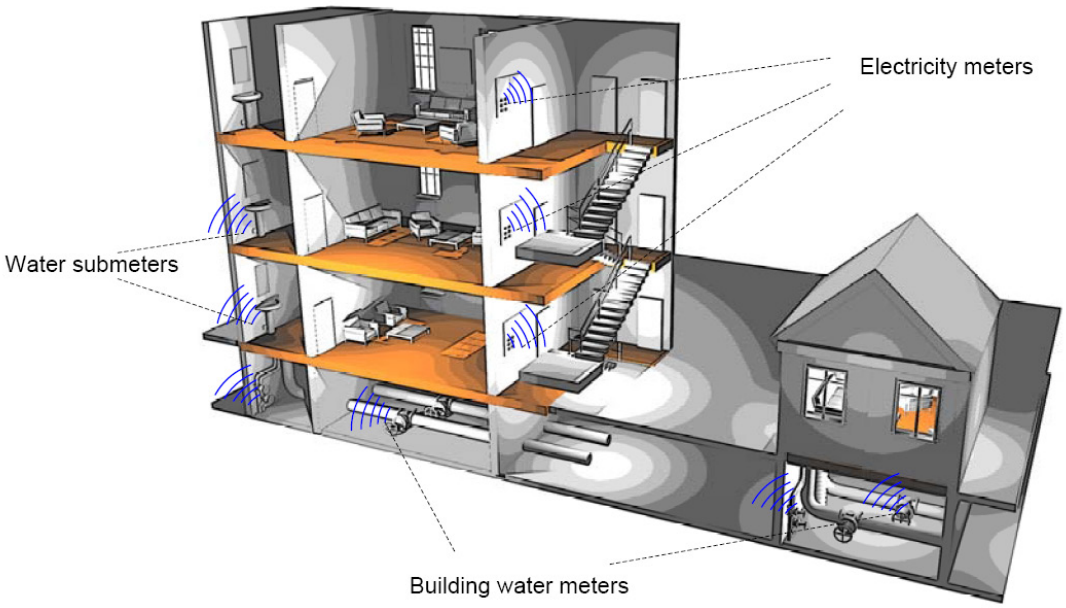


Figure 3.3: Examples for sensors in a smart building, from [5].

3.1.3 Smart Buildings

Smart buildings like they were presented in the scenarios from sections 1.2.1 on page 3 and in particular 1.2.3 on page 9 represent another environment adapting the ubiquitous computing paradigm.

In the currently 1.8 billion worldwide households [24], gradually many new context aware services will emerge, ranging from multimedia services to kitchen equipment and from lighting to presence monitoring. Nowadays there are already centrally controllable sensors detecting the building's context, such as electronic water and electricity meters, as shown in Figure 3.3, as well as sensors for carbon dioxide, temperature, light levels, weather conditions, electricity usage, noise levels and occupancy [28].

In addition, office buildings and public buildings are equipped with context aware services or inventory, for instance a lift can be controlled using the building's context. As there are less personal appliances in such buildings (like coffee machines, washing machines or multimedia devices), but more technical equipment (like elevators, escalators, automatic doors) than in households, it is fair to assume that both, households and office or public buildings will have plenty of context aware services to offer.

Real estate is much slower to take up advances in technology than mobile devices or vehicles, assuming an average durability of buildings of sixty years. In our time horizon of roughly twenty years only a third of all buildings will adopt ubiquitous technology. Taking into account that only 0.5 billion households (or roughly 30 %) have broadband internet connection today, growing slowly with about 1 % annually [24], we only consider these to be in fast technology adapting areas.

In twenty years' time, hence 50 % of all households will be in regions where ubiquitous technology already has entered buildings and only a third of all buildings will be sufficiently new to have recent technology built in. This amounts to 0.3 billion households. Taking into account also office and public buildings as well as retro-fitting of buildings, the number of smart buildings and households can be approximated with 0.5 billion in 2030.

	Number (in billions)	Services	Relevant Users	Relevant Context Attributes
Humans	7	5	1	35
Vehicles	1.5	2	1	3
Buildings	0.5	5	5	12.5
Rounded Sum (in billions)				50

Table 3.1: Calculation of the number of context attributes continuously monitored in a worldwide ubiquitous computing environment around 2030.

3.1.4 Summarised Context Usage

In the last sections we have identified context usage in three domains. This section will summarise the findings per environment and come up with a figure for context usage in total, as well as a figure of context usage per user of ubiquitous computing. Thereby only continuously running daemon services are taken into account, as they impose a higher burden for the ubiquitous infrastructure.

From what we have in the scenarios, most context aware services would depend on two or three contextual aspects, which are often a purposeful action, a status, or a target of one or several persons – high level context information. As relevant contextual aspects overlap however between different context aware applications, the remainder of this section assumes only one high level context attribute per service.

In section 3.1.1 it was already argued that four to five apps would be used on a daily basis and others from time to time. So we assumed here five services per user.

For personal vehicles, usually the time it is not used is significantly larger than the time it is used for driving. Therefore the number of continuously running services is only assumed to be two – for instance an alarm and an automatic opening service as described above which react to the context of the car owner.

Taking into account the scenarios in sections 1.2.1 on page 3 and 1.2.3 on page 9, it is realistic to assume at least five continuously running services for both households and office or other public buildings. More services are used on a on-demand basis. The smart buildings thereby do not only observe the context of a single person, like the smartphone or car owner, but multiple persons living in the household or working in the building. The average number of relevant users for a smart building is estimated to be five, as households in the most relevant regions usually contain on average four or less persons, but office and public buildings far more. The share of such buildings in the total number of smart buildings however is much smaller than the one of households which is why its influence only raises the average to five.

Following the grossly simplified prediction above, the calculation from Table 3.1 shows that in total fifty billion high level context attributes have to be continuously monitored worldwide by a ubiquitous computing system for its seven billion users. This means about seven high level context attributes per user.

3.2 The Dynamics of Context

The aim of this section is to see, how often context changes and hence how often context has to be inferred.

3.2.1 Activity Switches during a Working Day

As an example of a high level context aspect, the activity of a person is analysed. The reported change frequency of activities thereby also depends on its observation method.

There is an internationally used standard for activity classification for the "time use" surveys of many national bureaus of statistics. The revised standard for European time use surveys has been agreed upon in 2008 in [14] and represents "a minimum common denominator for coding the harmonised European Time Use surveys".

This hierarchical system defines three levels of activities with ten top level activities:

- Personal Care
- Employment
- Study
- Household and Family Care
- Voluntary Work and Meetings
- Social Life and Entertainment
- Sports and Outdoor Activities
- Hobbies and Computing
- Mass Media
- Travel and Unspecified Time Use

The ten top level activities are subdivided into 33 second level activities and 108 third level activities. These are to be further detailed as needed. This classification of mutually exclusive activities is also appropriate for use with Bayesian networks.

In a time use study in ten European countries with objects aged between 20 and 74 [13], people spend on average more than 45 % for Personal Care (including sleeping), more than 28 % for Employment, Household and Family Care and Study, both at home or not. While 5 % are spent travelling, 22 % are available for free time spent in the remaining five categories.

It is obvious that changes happen less frequently at the top level than at lower levels. But also changes between lower level activities of different top level activities are not so frequent. This can be explained as there is a clear temporal division between top level activities – for Personal Care and Employment caused already by the spatial separation of work and home and the need for sleep.

To investigate deeper into changes of lower level activities, this work analyses the Employment activity field.

Gloria Marks in [31] reports a study conducted with managers, financial analysts and software developers at high-tech companies. With a shadowing technique it was found that employees switch their activity, depending on their profession, on average every three to four minutes, i.e. fifteen to twenty activity switches per hour.

Similarly O'Conaill in [30] determines only external interruptions of two mobile professionals for whom communication formed a central part of their job. Both "were shadowed with a video camera for a full working week." She has registered eight interruptions per hour, hence nine different activities per hour. The difference to Marks' experiment is

probably explained by the fact that the latter also observed internal interruptions which amount to about 50 % [31].

In a self-reporting experiment observing two scientists, this work has analysed the number of activity switches during a working day also. Although it is less exact than a shadowing approach, it rather focusses on important activity switches. They reported on 79.4 hours with 331 activities during nine working days in total. As expected, with the different methodology this approach yields a significantly lower number of activities with only 4.2 activities per hour.

3.2.2 Change Frequency of Lower Level Context

Important for the frequency with which high level context like activity changes is the lower level context it depends on. For that reason location changes shall be considered in the following paragraphs.

The principal mode by which people move is walking, one of the slowest but most frequent change of position. It is agreed to be healthy to take about $10,000$ steps a day. This number is usually reached by healthy adults and even elderly still take between $4,000$ and $8,000$ steps [3]. "Consistent evidence supports that 30 minutes of at least moderate-intensity walking is equivalent to $\sim 3,000$ steps" [16]. This means a little bit less than two steps or roughly 1.5 meters movement per second.

Location changes can also be considered at a higher level, for instance the changes of locations in terms of rooms during a working day. A self-reporting experiment has found that the author visits about twenty different locations a day, counting the way to work, to the canteen or to the supermarket as only one location each. During the day the author changed more than 80 times between the rooms and locations, i.e. during the 16 hours awake this means five different locations per hour or one change every twelve minutes.

The frequency relevant for location updates depends on the precision necessary for the applications using it and the moving speed of the tracked object. Current sensors for location include among others receivers for the *Global Positioning System (GPS)* outdoors, *Ultra Wideband (UWB)* systems or Wi-Fi/mobile-radio fingerprinting systems for indoor positioning or *Inertial Navigation Systems (INS)*. All systems have different update frequencies, ranging from 1 Hz (e.g. GPS or Wi-Fi signal strength measurements) over 10 Hz (maximum mode for UWB sensors) up to over 100 Hz for some INSs.

Although obviously very high update rates for low level context exist, this work assumes that a reasonable update frequency lies around 1 Hz. It coincides roughly with the frequency of a step or the delay of human reaction time of roughly 1 s.

3.2.3 Context Inference Frequency

In the last sections it was shown that high level context like activity usually changes every few minutes. The frequency with which it has to be inferred however depends mainly on the change rate of its influencing low level context. Significant changes in low level context can happen as often as once a second, for instance with a step out of a door.

There are two general ways for context inference for the continuously monitored seven high level context attributes:

(1) Inference upon change of low level context:
Every time one of the impacting low level context changes, high level context is re-inferred. Similar to the well known "butterfly effect" [20], all low level context can have impact on your high level context. This can result in very high update rates: in the subsections of section 3.1, it was shown that hundreds of sensors are

constantly tracking changes in a user's domain, be it his smartphone, his home, his office or his car. Assuming only 1 Hz as the rate of all (un-synchronsied) sensors, this could result in hundreds of inferences per second.

(2) Inference with fixed delay:
In the previous section, 1 Hz was assumed to be an acceptable delay for context inference in many cases, as it coincides with human reaction time. (Depending on the actual application, the acceptable delay can also be significantly longer or shorter.) Hence a plausible approach would be to infer once per second taking into account all current context. In order not to exceed the one second delay, all context information would have to be transferred to the inferring CPU and calculated there within this second – practically impossible for hundreds of sensors, possibly large distances and network interfaces overloaded by the seven billion assumed users worldwide.

Both approaches are obviously inappropriate for the continuously monitored context attributes and the immense number of future ubiquitous computing users, both for centralised approaches as for a distributed inference architecture.

3.3 Requirements for Context Inference

The abstract target of context inference is always to infer consistently and correctly new information from the existing one in the shortest possible response time. The main challenge to realise this are the large dimensions of context inference in ubiquitous computing. It affords solutions for scalability, both in terms of users and in terms of low level context, such as sensors, incorporated in high level context inference.

Given the situation presented in the scenarios and detailed in the last sections, the following requirements are therefore considered necessary. They are fundamental for adaptive and tractable Bayesian context inference.

1. Decentralised, modular inference:
   Realistically central servers cannot perform context inference for billions of users taking into account hundreds of sensors per user. Inference has to be performed within the user's smart space. Information coming from remote devices or smart environments should be preprocessed at their origin as far as possible. Therefore inference has to be modular, encapsulating context from different environments and enabling to plug the different modules together for final inference.

2. Reduction to relevant input:
   To shorten inference time, the amount of low level context taken into account has to be reduced to a minimum. Only situational information relevant for the respective high level context must be taken into account. Modules that are not relevant can be neglected.

3. Demand driven inference:
   The approaches presented above show a very frequent need for re-inferring context. While continuous context inference has to be offered for situations and context attributes which absolutely require it, it has to be reduced to the minimum possible. Only when there is a service requesting high level context, it should be inferred. If not requested explicitly, context should not be inferred continuously when not necessary.

4. Support for different characteristics of context:
   Different context aspects change with different frequencies, usually lower level context more frequently. Higher level and less frequently changing context should be inferred independently from low level context. Similarly, not all context will be stemming from Bayesian inference approaches, but should still be incorporated in high level context inference, as far as possible. This can be realised with different inference modules.

5. Access control and privacy:
   As context has to be computed partly locally and is requested from remote devices and even other users, access control mechanisms have to be embedded to protect the information owner's privacy and to only disclose the context to which the owner has granted access.

6. Adaptability:
   The requirements above require that less influential information is neglected. But as soon as previously omitted information becomes relevant to the system it should be considered. Thus, a characteristic that is essential due to the dynamic nature of context-aware computing systems is the adaptability to changes. The nature of the changes could be inclusion of new sensors for already considered context aspects or even new aspects. The realisation thereby has to be efficient in time and computer memory in order to guarantee timely continuation.

Chapter 4

State of the Art for Tractable Context Inference

As has been seen in the section 2.2, from page 20 onwards, a lot of work has already been done on context inference. A particular focus of this work is the computational tractability of context inference. Therefore this chapter shall illuminate the state of the art for those fields that are most relevant for tractable Bayesian context inference .

For the inference of high level context numerous aspects of lower level context can be involved, e.g. such related to the user's behaviour, information about the current status of the ubiquitous system, as well as the user's interaction with others. Therefore not only general approaches for scalability (in section 4.1) and modular Bayesian techniques for context inference (in section 4.4) are presented, but also the existing work for two lower level context aspects of outstanding importance, location (section 4.2) and human motion related activity (section 4.3).

4.1 Scalability of Context Aware Systems

Different areas have to be scalable to enable large scale context aware computing, in particular context management and service management. According to Neuman in [260], scalability has three dimensions:

- the numerical dimension: the number of users, context sources and services,
- the geographical dimension: the distance between the farthest interacting nodes,
- the administrative dimension: the number of administrative organisations involved – hence a measure of the heterogeneity.

Management of context aware service in large scales has been dealt with, e.g., by Barratt et al. in [247], by Kerry in [256], Storz in [269] and in particular by Buchholz in [249]. They all exploit the characteristics of *grid services*, a semantic extension to web services, to reach global scalability. Buchholz proposes an integration layer to deal with the above three dimensions of scalability and distributes the context aware services with existing content delivery networks.

Context Provisioning or context management also has to be scalable in all three dimensions mentioned above. Buchholz has given a classification of different approaches and related it to geographical scalability as can be seen in Figure 4.1. The context management systems presented above in section 2.1.3 on page 18 all follow infrastructure based service-centric approaches. They provide homogeneous interfaces for context storage and

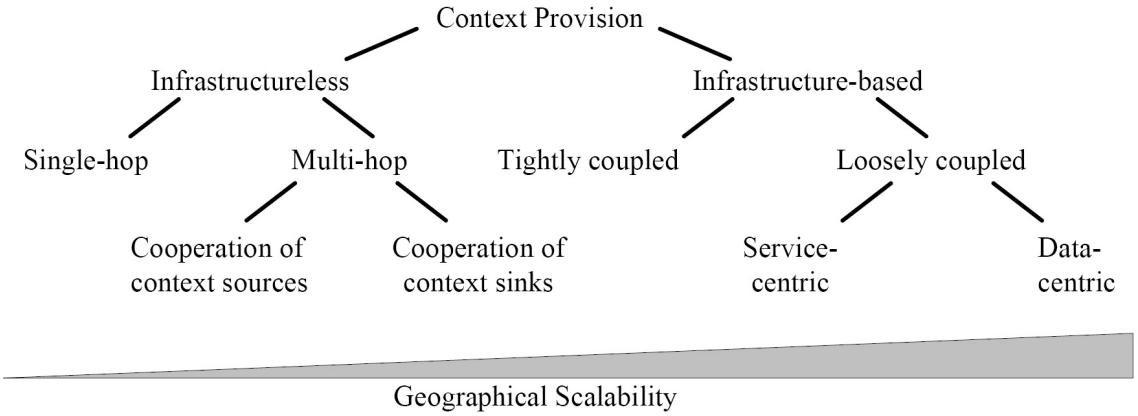


Figure 4.1: Classification of context provision approaches after Buchholz in [249].

retrieval from sensors. Dey's Context Toolkit [10] was one of the first context management frameworks to encapsulate sensors in so called widgets that provide a unified interface for context aware services to decouple context sources from the framework. With this approach Dey et al. reduced the heterogeneity and address the administrative dimension of scalability.

A loosely coupled system is a prerequisite to scalable context provisioning. In addition, the infrastructure has to be distributed, also. This has developed more and more in the CMS prototypes presented in section 2.1.3, for instance SOCAM, CroCo, CDDBMS, the Persist CMS or the federation of multiple context brokers with an asynchronous event-based publish-subscribe paradigm by the C-Cast project in [25]. The concept of personal smart spaces is an approach to both, geographical and numerical scalability in which local sources of information are associated with a local smart space. The number of local information is smaller and easier to handle with this approach. All smart spaces interchange information and route context requests among each other, to reach consistency and completion of all requests.

The numerical dimension of scalability in context provisioning implies scalability of context creation, hence in particular reasoning and inference. As reasoning usually involves significant computational resources, and an acceptable response time of a few seconds at most must not be exceeded, it is necessary to provide measures that ensure that large numbers of reasoning requests can be handled without significant delay. Different ways for scalable semantic (description logics based) reasoning have been proposed, for instance in [268], [248], and [259]. These approaches relate more to large reasoning domains than to large numbers of reasoning requests – which makes sense for semantic reasoning, as relations being found do not change too frequently.

Scalable approaches to context inference, however, must be realised by a combination of *minimising* the number of relevant requests by context management and *fast* inference including only the *relevant* and *necessary* information. No general approach for this has been found in the current literature. This work will propose an approach for that goal in chapter 5.

4.2 Positioning and Location Management

Context inference is almost unthinkable without the inclusion of location information. While its importance is explained later in section 5.2.1, this section shall give an overview

over the research undertaken so far to provide exact information about the current location in all environments and to manage such data.

4.2.1 Positioning

Positioning is the detection of the current location of an entity. The manifold approaches depend among others on the environment. In particular methods which work very well in outdoor environments are often not applicable indoors, where the infrastructure and therefore the available information is different. Moreover, the positioning requirements are often different outdoors and indoors. Therefore the following subsections shall present the general approaches for positioning outdoors and indoors.

4.2.1.1 Outdoor Positioning

Outdoor positioning today is dominated by *Global Navigation Satellite Systems* (GNSS), of which to date GPS is fully operational and the GLONASS, Compass and Galileo systems in development. In areas of adequate visibility of GNSS satellites, the use of dedicated portable navigation devices or cell phones and PDAs equipped with GPS receivers has increased dramatically over the last few years providing personal navigation in vehicles and also pedestrian navigation in cities and recreational environments.

A GNSS allows receivers to determine their location (longitude, latitude, and altitude) using time signals transmitted by radio from multiple satellites. Receivers calculate from the received signals the time, longitude, latitude and altitude by trilateration, assuming a line of sight connection to the satellites.

The precision of a GNSS systems varies usually between a few meters and a few hundred meters, depending e.g. on the visibility of the satellites and possible multipath effects. This precision is usually sufficient for outdoor purposes, e.g. car navigation. With such *receptive* positioning [178], where the mobile device computes its own location from ubiquitously distributed signals, the mobile device is independent from servers, which calculate the position in the complementary *transmissive* positioning, from active communication causing costs, and has immediate privacy.

As the high infrastructure efforts are already spent and running cost come from a different budget, GNSS receivers are cheap, small and relatively energy efficient, coming with own CPU for the positioning, GNSS is the perfect choice for outdoor positioning. The CPU is not burdened with extra calculations, no running expenses are generated and the obtrusiveness of a mobile device is not impacted.

4.2.1.2 Indoor Positioning Systems

Positioning in indoor environments is problematic for two reasons: firstly, the difficulty of GNSS reception results in much greater deficiencies in accuracy and availability. Secondly, the desired accuracy for meaningful location dependent services is often much higher than outdoors – the difference of one meter (about one step) can make the difference if a user is in one room or another. The accuracy indoors quite always depends on the requirements of pedestrians, while outdoors positioning is often used in car traffic. The differing mean velocities imply the different accuracy requirements.

Indoor positioning methods can be based on infrastructure or rely on additional sensors worn by the user. An overview over current methods has been given by the author et al. in [151]:

Infrastructure systems fall into two categories: dedicated wireless arrangement (for example the Cricket system [181] or Ubisense [182]), or adaptation/usage of existing communications infrastructure like Wireless LANs [195, 167, 184, 183].

In order to achieve truly ubiquitous personal positioning and navigation indoors and to maintain tractability, a successful approach needs to be *as autonomous as possible*, requiring a minimal amount of additional standardisation and dedicated infrastructure, whilst building on the rapid advances in portable data processing and sensors. Additionally, transmissive techniques will always remain questionable in terms of privacy considerations.

The aim has to be hence to use the existing infrastructure which is advantageous in terms of costs and therefore also general applicability. In particular in buildings in which people require personal navigation, such as office buildings, airports, or large public buildings, there now exists a dense installation of Wi-Fi infrastructure, often operated by different operators, for example in airports, public buildings, and company premises, which eases fingerprinting, as more access points are available. In an approach using existing infrastructure, moreover the position can be calculated locally on the mobile device just like for GNSS receivers.

Two general approaches realising these requirements shall be presented in more detail. On the one hand WLAN positioning exploiting the existing infrastructure, and on the other hand inertial pedestrian navigation which adds user worn sensors to the location determination. In both cases, no extra infrastructure is required and the location is calculated on the user's own mobile device.

WLAN Positioning

There are two primary methods for location determination from Wireless LAN [161, 160] based around Received Signal Strength Indicators (RSSI).

This first method is based on propagation models using estimated degradation of signal strength over distance in space from the known location of access points and their transmit power. Typically the indoor environment is modelled as established by the COST 231 standards [147]. The calculated distances are then usually used to estimate a location through trilateration [167].

The other method is empirical and known as *Location Fingerprinting*, *Wi-Fi Fingerprinting*, or *WLAN Fingerprinting*. It stores pre-recorded calibration data in order to generate a radio frequency map of a building. The location of a mobile station can be estimated like that by correlating its RSSI measurements with the radio frequency map constructed and stored. This method is widespread today and has been used for instance in [196, 195, 194, 142].

Location fingerprinting algorithms can be classified into two main categories: deterministic and probabilistic. Deterministic algorithms are using a set of constant location fingerprints, which includes mean vectors and standard deviation vectors of RSSI. Probabilistic algorithms model the location calibration points with RSSI probability distributions.

The advantages of this approach are the following: The wireless infrastructure information, such as the exact location of the access points, is not required. Only the reference location of the RSSI samples collected during the map construction is needed. Furthermore, fingerprinting can operate using only one active access point, although it will likely produce worse results over large areas. It typically results however in higher accuracy over methods based on propagation models [194, 142].

However, fingerprinting requires time to train and results are dependent on the time of training as obstructions created by differing numbers of people affect the calibration sample data. Hence, the main drawback of this method is the generation and maintenance of the radio frequency map which can be time-consuming and expensive when performed over wide areas.

Inertial Navigation for Pedestrians

The use of inertial sensors is becoming widespread for pedestrian navigation, especially for indoor applications. Basically two approaches can be distinguished:

- The *pedometer-approach* employs an accelerometer for detecting individual steps whilst the stride length and stride direction are themselves estimated using additional sensors, such as GNSS or a priori information. Given a detected step, its length and its direction, a person's position can be determined by dead-reckoning [152, 171, 146]. Other methods have been studied in [170].

Figure 4.2: A foot mounted inertial platform for pedestrian navigation used at DLR.

- The latest approaches are based on full *six degree of freedom (6DOF) inertial navigation*. A foot-mounted 6DOF strapdown inertial platform comprising triads of accelerometers and gyroscopes, see Figure 4.2 for an example, is used to dead-reckon via a conventional strapdown navigation algorithm. An EKF (cf. section 2.3.6.2) runs in parallel to the strapdown algorithm. Rest phases of the foot detected in the accelerometer signals trigger (virtual) zero-velocity measurements which are used to update the filter, so called *Zero Update* (ZUPT) measurements. With the regular ZUPT measurements the drift errors which accumulate in the strapdown solution can be estimated and corrected [150, 153, 143, 156]. It was shown in [150] that this *pedestrian dead-reckoning (PDR)* approach can achieve very good performance even with today's low-cost micro-electro-mechanical (MEMS) sensors, because the ZUPTs are so frequent that errors build up only slowly during each step the pedestrian makes.

The benefit of combining foot-mounted inertial sensors with non-linear map-matching techniques or additional non-linear or non-Gaussian sensors typically used in an indoor scenario with particle filtering techniques (cf. section 2.3.6.3) has been presented in [168] and [192]. Comparable results were shown for a 2.5 D environment in [193]. The benefit of integrating a pair of platforms that are mounted on each of the pedestrians' feet respectively has been studied in [169]. In these sensor fusion approaches foot displacement

and heading change values from the foot's PDR filter are computed at each step and are exploited as measurements within a higher-level main fusion filter.

4.2.2 Location Management

Next to the detection of an entity's location, also the representation, sharing and reasoning about is is important to allow for a maximum benefit. There is a lot of related work on these topics. The basic issue is the representation in a location model, as efficient models and architectures are a necessary prerequisite for efficient customisation, adaptation and inference [173].

Domnitcheva identifies as the major challenges for location modelling: scalability, privacy (e.g. blurring the current location by probabilities, like in [141]) and spatial awareness. In [148], she gives a comprehensive overview of the developed models. She classifies them in particular into *geometric* (using absolute coordinates) and *symbolic* models (using names, in particular also addresses). While the high resolution of geometric models often overburdens resource limited devices, symbolic models often suffer from an unknown, not shared terminology and difficulties to relate locations to each other. *Hybrid* models (also called semi-symbolic or combined models) try to overcome these issues but bear a high computational complexity themselves.

Pure location based services like Whrrl[1], Google Latitude[2] or Plazes[3] use geometric location models. Hence they are to date restricted to coordinates or addresses on a map. These services however do not have the need to reason about locations. Links between absolute and symbolic locations are established by *Geocoding* (symbolic to absolute) or *Reverse Geocoding* (absolute to symbolic location). Reverse geocoding hence plays an important role for translating precisely inferred absolute location to symbolic location for context reasoning. Only recently a higher number of reverse geocoding providers has come up, e.g. with the reverse geocoding API of Google[4], or the Nominatim[5] tool for OpenStreetMap[6]. These services however are not directly applicable to produce the symbolic location necessary for context reasoning, as they neglect uncertainty information inherent to inference of absolute location.

An important factor for reverse geocoding services is the efficiency of processing requests. Therefore they are often based on *Geographic Information Systems (GIS)*. Efficient location management is one of the targets of a GIS. They capture, store, analyse, manage, and present data that are linked to location [154]. GIS therefore merge approaches from cartography, statistical analysis, and database technology. Particular remarkably are here efficient store and search structures, like the *R-Tree* [158].

Research on reasoning with locations has produced large practical and theoretical works, mainly known as *spatial reasoning* or *location reasoning*. Varzi explains in [189] that spatial reasoning involves theory on parthood relations (*mereology*), qualititative spatial relations like continuity and contiguity (*topology*) and the theory on relations between an entity and the spatial region it occupies. Myers' location reasoning theory [174] extends such considerations also to reasoning about movements. She defines deductive rules which can be classified as a subset of description logics, with which facts about location can be

[1] `http://whrrl.com/`
[2] `http://www.google.com/latitude`
[3] `http://plazes.com/`
[4] `http://code.google.com/apis/maps/documentation/geocoding/`
[5] `http://wiki.openstreetmap.org/wiki/Nominatim`
[6] `http://www.openstreetmap.org/`

added or retracted, "initiated by either operators (in the case of a planner) or sensors (in the case of a situated reactive system)" [174]. Practical work in the same direction, symbolic locations semantically related in some kind of knowledge base, has been published among others by Guggenmos in [157] with a server based location management system managing several ontologies for fixed semantic annotations, by Hu and Lee in [159] who store locations and exits in maps usable in mobile environments, or by Kolomvatsos in [166] who has a system for semi automatic instantiation of a location ontology, based on annotated data. They all perform semantic reasoning that is able to reveal parthood, inheritance, transitivity and similar relations between locations.

Approaches with hybrid location models contain among other those of Becker [145] and Narayanan who formalises in [175] the translation of geometric, raw locations into states of spatial realms, i.e. symbolic location. Malkani's work in [173] proposes the integration of service discovery with positioning systems. In his approach a server translates locations of different positioning systems into each other. The server contains a hierarchical location model and manages the presence of mobile devices in leaves of the respective location tree. Pfeifer's *Redundant Positioning Architecture* [178, 177, 180, 179] and Zündt's *Distributed Community-Based Location Service Architecture* [197] in addition focus on the exchange of location information between different autonomous domains.

Most context frameworks have decided for an ontology based context modelling approach like [38], [42], or [18]. Locations are therefore only represented as symbolic names, reasoning is limited here depending on the semantics encoded in the ontologies and does not allow for general high level context inference. Projects like IST Daidalos[7] and ICT Persist[8] used symbolic locations for machine learning techniques, but had to neglect relations between location names or a close link to highly precise location estimation techniques. Krause et al. in [78] recognise high level activity patterns from a couple of sensors. For location they use GPS coordinates and locations of Wi-Fi Access Points in range. They cluster coordinates that are closer to each other than 50 m using the k-Means algorithm based on the Euclidean distance and choose along fixed rules between both sensors. Every cluster is used as a hypothesis in the estimator later on. Their semi-unsupervised approach is computationally very costly and completely neglects symbolic locations. It is not usable for location queries, not aware of borders between locations and not usable for untrained context inference.

4.2.3 Discussion

The current state of the art in positioning yields very good results, even in indoor environments. There is however room for improvement with regards to the scalability of the approaches. Receptive approaches which do not afford network communication nor extra investments in the infrastructure, such as Wi-Fi fingerprinting, can use the existing Wi-Fi infrastructure. To obtain best results there, Wi-Fi fingerprinting requires time intensive training. To reduce this complexity, its results can be fused with body worn sensors. This however increases in the presented approaches the computational complexity with the resource intensive particle filtering approaches.

The concepts for location management seem quite mature. They allow for heterogeneous sources, different formats and exchange across different autonomous domains. The

[7] `http://www.ist-daidalos.org/`
[8] `http://www.ict-persist.eu`

existing approaches however largely neglect the process of providing the location information to manage. As precise positioning techniques exist, this reduces to the respective provision of symbolic location information maintaining the uncertainty information of the positioning process.

While reasoning about locations and their interdependencies exist with the current approaches, research on the generic inclusion of location information in high level context inference is missing.

4.3 Activity Recognition

The current activity of a user has a similar importance for context awareness as his or her location. This research topic has been one of the hottest in the field of ubiquitous computing in the recent years. To be useful for generic context inference, the approaches have to work in real-time without significant training effort and under all environmental circumstances, while keeping the computational cost at the necessary minimum. A prerequisite for the user acceptance of a activity recognition system moreover is its affordability and its unobtrusiveness.

A synopsis of the current work in activity recognition with a special focus on human motion related activities, following the overviews by the author and Vera Nadales in [209] and [237], shall be given in the remainder of this section. First the different recognised activities shall be illuminated, before the sensors and the recognition techniques are presented and discussed regarding their relevancy to the demands placed in the previous paragraph.

4.3.1 Activities to Recognise

The recognition of a user's current activity is relevant for health care and ambient assisted living, as well as for entertainment applications and video games, mobile context advertising or work monitoring, as has been shown in the scenarios presented in section 1.2 on page 3.

The relevant activities thereby refer partly to human motion related activities and partly to higher level activities, often called *activities of daily life* (ADL). ADLs are more difficult to recognise as they consist often of different steps and depend stronger on environmental settings. Motion related activities however are more often necessary, they are also important input for the recognition of ADLs.

Most of the related work on activity recognition to date is focussed on inferring human motion and posture related activities [229, 212, 198, 223, 215, 227, 236, 216, 224, 217, 238] rather than high level activities like in [232, 130, 214, 206].

There are a number of motion related activities studied in the literature [212]:

- Transit activities like *walking, walking upstairs and downstairs* or *running.*
- Abrupt activities like *jumping* or *falling.*
- Sports activities such as *jogging, cycling, rowing, callisthenics* or *martial arts moves.*
- Actions such as *open door* or *close door.*
- Postures like *standing, sitting* or *lying.*

4.3.2 Used Information Sources

The information sources used for activity recognition are an important factor for the usability of the system. Multiple kinds of sensor modules have been employed in the literature. Their selection depends on the available infrastructure, the activities under investigation and also the applied recognition technique. Huynh in [212] includes a detailed review of the sensors used. It contains among others:

- Accelerometers [229, 212, 198, 223, 215, 227, 200, 242, 220, 243, 201, 202, 236, 204, 216, 224, 217].
- Gyroscopes [231].
- Force sensitive resistors [227, 232].
- Foam pressure sensors to measure respiration rate [203].
- Digital Compasses [212].
- Physiological sensors such as oximetry sensors [226], skin conductivity [239] and heart rate sensors [227, 223, 215].
- Temperature, humidity and barometric sensors [212].
- Light sensors [220, 227, 212, 201].
- RFID tag readers [232] and RFID tags placed on objects.
- Microphones [212, 220, 201, 214] for both, the user's voice and environment specific noise.
- Video with feature recognition [221].
- Fiber optical sensors to measure posture [207].
- Stretch sensor, as it was used in [208] in a jacket.

Accelerometers in various forms are the most common sensor used in the related work of activity recognition. They are used at different parts of the human body, as orthogonal combination of two or three accelerometers for different dimensions, and also in combination with other inertial sensors in *Inertial Measurement Units* (IMU).

A reason for their widespread use is its close relation to motions and the fact that the most investigated activities are related to human motion. As accelerometers and IMUs can be built with cheaper and cheaper *Microelectromechanical systems* (MEMS), the cost factor and the availability in mobile devices like laptop computers, smartphones and cameras are further comforting factors.

The majority of the considered related work has used accelerometers spread across multiple locations of the human body [199, 200, 198, 223, 227, 220, 243, 201, 231, 236, 224, 217], as shown in Figure 4.3. The work of Choudhury in [205] has demonstrated that under certain conditions, using multi-modal sensors at one single location of the body can offset the information lost from other parts of the body. Also the research in [232, 202, 215, 212, 229] uses several sensors in one single location.

Also further information like maps, GPS information, bus schedules, people's habits or the use of objects are partly used to improve the results (e.g. [229, 232, 235]).

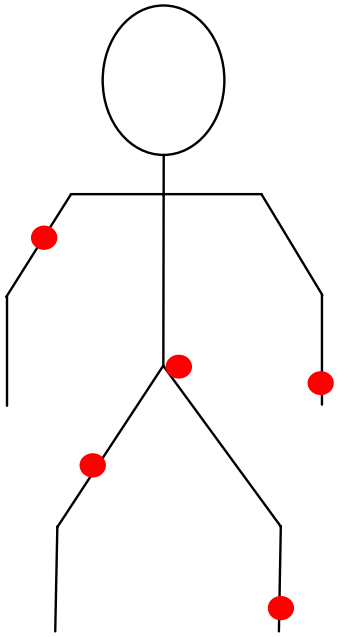


Figure 4.3: Distribution of sensor boards (marked in red) with accelerometers across the human body in the work of Bao [199]. Five accelerometers are placed and secured on the left arm and thigh, the right wrist and ankle and the right side of the person's hip.

4.3.3 Recognition Techniques

The evaluation of the collected sensor data is a typical classification problem which can be solved with the methods outlined in section 2.2 on page 20. The choice of the classification technique depends thereby usually on the available data and the requirements towards outcome and training phase.

The considered related work covers a large range of the classification techniques presented in section 2.2. For instance decision trees are used in [200, 202, 201], kNN classification in [220, 201, 200, 227], SVMs in [221, 202], neural networks in [216, 221, 227, 229]. Other examples of classifiers used in the related work are probabilistic techniques in [231, 201, 243, 220, 200, 221] and, in particular, particle filters in [230] or [228].

A crucial difference between the classification techniques is the effort necessary for training them. Training a classifier can be *unsupervised* or *supervised*, be based completely on machine learning algorithms, consider human imposed restrictions, or completely be human defined.

Unsupervised training identifies repetitive patterns in a so called *training set*, e.g. groups of data with little difference in specific measurement data [212]. They have been used among others in [219] and [225]. A classifier trained completely unsupervised however cannot assign semantic information to the identified clusters and is therefore inappropriate for most applications of context awareness.

Semi-supervised approaches (used e.g. in [212, 234, 233]) are methods that can be applied when parts of the available data are labelled, while for other parts no labels exist. Supervised learning requires a completely labelled data set. As in these approaches the labels define the identified clusters, they are more appropriate for activity recognition. The main disadvantage is the effort to label the data.

4.3.4 Discussion

The comparison of the different inference approaches is difficult. The classifiers are trained on different data sets, the data sets and the evaluation are performed under different environmental conditions (see the work of Stephen Intille on their relevancy in [213]) and the evaluation results are reported in different measures.

Activity recognition can be evaluated under the following conditions:

- Laboratory conditions: In these, the best recognition results will be achieved, but they are less reliable under real conditions.
- Semi-naturalistic conditions: Naturalistic conditions are simulated. For example, [221] prepared an artificial living room in a lab to collect the required data. Examples of related work on activity recognition under semi-naturalistic conditions are [198, 227, 200].
- Naturalistic conditions: In this case the information is collected during a normal day of a person, under real conditions. Very few of the related work collected data under naturalistic conditions [215, 236], as in particular the verification is more difficult and would have to resort to person or video based shadowing techniques.

The same distinction holds for the acquisition of the test data, the more natural the data set for training the classifier is, the better it will perform under naturalistic evaluation.

Also measures of the quality of activity recognition systems, their accuracy, differ. An overall accuracy value (like in [198, 227, 221, 242, 231, 204] is not meaningful with regards to the quality of the recognition of every single activity and the weighting of the different activities. Expressing the accuracy with precision and recall values for every activity like in [218] is therefore more expressive. Some works (like [200]) provide even the full confusion matrix from which precision, recall and other measures can be computed.

Moreover, the referred evaluation results depend on the evaluation methods. Evaluating a system based on the data of the person whose data was used to train the system like Bao's work in [200], only reports on the ideal performance. To make these results transferable, every user has to train the system before it can be used – which is a lengthy and troublesome process with regards to labelling the training data.

A selection of related research with apparently good results is shown in Table 4.1, together with the specification of how the data set was acquired, which sensors they use and where they place it.

The best and most stable recognition accuracies for human motion are reached when sensors are distributed over different positions on the body, which makes such recognition systems however more obtrusive and less practically usable.

For the recognition of activities of daily life, the best performing system from [221] relies on video analysis, which is not viable for ubiquitous computing. Also the other approaches work reasonably well, the overall recognition accuracy however is too low to be completely reliable. Moreover the system and the sensor placement needs to be tailored to the specific high level activities to be identified.

Therefore it makes sense to separate the recognition of human motion and higher level activities. Once there is an unobtrusive and very reliable system for the recognition of motion related activities, its results can be used for any type of high level activity inference. Current research on this topic however has not been identified.

Classifier training of those works which have given insight in their publications can be improved. All approaches have used standardised training libraries like the Weka toolkit

Ref.	Activities	Information Sources	Sensor Placement	Data set	Recognition Accuracy
[221]	ADL	Video	-	1 person (S)	95%
[199]	ADL	5 Accelerometers (2D)	Left thigh, right ankle, both arms	20 people (S)	84.26%
[201]	ADL	Accelerometer, light sensor, microphone	Chest, wrist & shoes	7 people (L)	$\geq$ 90%
[231]	ADL: bike repairing	Accelerometer	Torso, left & right sleeve, left upper & lower arm & left hand	3 people (L)	82.7%
[227]	Human motion, ADL	Accelerometer (3D), compass, ambient light, force sensor	Right thigh, necklace, right & left wrist	13 people (S)	90.61%
[242]	Human motion, ADL	Accelerometer	Dominant wrist	7 people (L)	92.86% +/- 5.91%
[229]	Human motion	Accelerometer (2D), GPS	Pocket	10 people (L)	85% - 90%
[198]	Human motion	Accelerometer (3D)	Chest & thigh	5 people (S)	89.3%
[236]	Human motion	Accelerometers	Wrist & thigh	1 person (N)	86% - 93%
[216]	Human motion	Accelerometer (2D)	Left upper leg	6 people (L)	42% - 96%
[224]	Human motion	Accelerometers	2 at hip	1 person (L)	83% - 90%
[217]	Human motion	Accelerometers	2 at thigh	8 people (L)	92.85% - 95.91%
[204]	3 Kung Fu movements	Accelerometers	2 at wrist	1 person (L)	96.67%

Table 4.1: Selection of the state of the art on activity recognition, adapted from [237]. The considered activities of daily life contain among others bathing, dressing, opening a drawer or different steps of repairing a bicycle. The data set column indicates the conditions of the data collected: under laboratory (L), semi-naturalistic (S) or naturalistic conditions (N).

[137]. Specifying limitation and imposing human knowledge like causality could further improve the results and has an impact on the recognition time.

Concluding, research should focus on a highly reliable system for the recognition of human motions, comprising only one sensor module. To achieve general applicability without extra training and optimal evaluation time, neither the selection of training data nor the optimisation of the training process must be neglected.

4.4 Modular Bayesian Networks

Bayesian networks have been identified as a suitable means for context inference in section 2.2 on page 25 and their theoretical background has been presented in section 2.3 from page 27 onwards. The problem to be solved for its applicability for context inference is the computational complexity of the probabilistic inference in them. As a core requirement, section 3.3 on page 58 therefore has identified the need to break BNs down into smaller BN modules.

Several proposals to divide large BNs into such BN modules exist in the literature. Along with a possibly reduced inference time, the break-up eases storage, re-usability and specification of networks. The remainder of this section shall present some of the proposed approaches along with the respective inference methods, before their applicability for context inference is discussed.

4.4.1 Approaches and their Inference Methods

Already in the early 1990s, a first approach for sectioning Bayesian networks has been proposed. Throughout the last two decades more ways have been proposed tailored to specific application needs and with dedicated inference approaches.

The following section shall report on the most relevant work in this area and analyse its applicability for tractable context inference.

4.4.1.1 Multiply Sectioned Bayesian Networks

The first published approach to splitting Bayesian Networks has been published by Xiang in 1993 [273]. The core assumption of his *Multiply Sectioned Bayesian Networks (MSBN)* is that within close temporal proximity, always the same parts of a large BN receive evidence and are queried. As frequent propagation of evidence in a large BN is inefficient, the MSBN concept focusses on modifications of only parts of the BN.

To this end, the division of a BN into localisation preserving partitions, Bayesian subnets are introduced. Different partitions can share variables which d-separate them. For every random variable there has to be at least one partition which contains the node with all its parents.

Partitions in MSBNs have to form a tree structure, the *partition tree*. The shared variables of subnets have to be in all subnets on the path between them in the partition tree. The partition tree is constructed based on the known, completely specified BN which is partitioned exploiting the known d-separation properties. Similar to a junction tree in PPTC (see page 37 and [86]), this partition tree contains *d-sepsets* containing the d-separating *d-sepnodes* of two adjacent subnets. The JPD of a d-sepset is the only information necessary to be exchanged between different Bayesian subnets. Hence, d-sepsets act as interfaces between pairs of subnets.

A first step to the construction of the partition tree is the construction of junction trees of the identified partitions. The set of subnets of the MSBN is then moralised

and triangulated into a set of *morali-triangulated* graphs, a process using the knowledge about the complete network. As a next step the now existing *junction forest of cliques* is inter-linked by the d-sepsets.

An example of a linked junction forest of cliques is shown in Figure 4.4. The subnet junction trees are called Γ^1, $\Gamma^2$and Γ^3, links are represented by bold lines, while the rest of the edges represent separators between cliques belonging to the same junction trees.

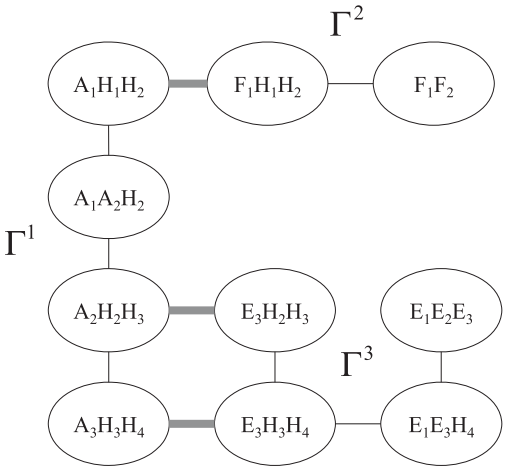


Figure 4.4: Example junction forest [273].

Initialisation of belief potentials and inference between different subnets roughly follows the process of PPTC.

The above assumptions and construction rules allow for independence of the subnets, therefore also for partly independent modification and evaluation [272]. In particular, the subnets can run their inference processes asynchronously in parallel. This option increases the performance of the system and reduces security risks as only resulting probabilities of networks have to be passed by remote communication and not whole networks that may reveal personal information.

All in all, MSBNs provide an effective and exact approach for distributed inference but also impose a set of constraints. In particular the partition tree organisation prevents subnets from communicating arbitrarily with another agent, which would constitute a serious limitation for context inference.

4.4.1.2 Network Fragments

Laskey and Mahoney introduced the concept of *Network Fragments* in 1997 for the domain of military situation assessment, i.e. for determining who the actors are, where they are located, what they are doing, and what they are likely to do in the future [258]. To this end, a fixed model (with variables and causal relations assumed to be the same for every user or situation) is not appropriate, as relevant variables and their probabilistic relationships vary.

A network fragment organises RVs and their conditional probabilities. Fragments are objects which can be organised in hierarchies of related objects, share structure and behaviour. They distinguish between input (parents only outside the fragment) and resident variables (parents only inside the fragment). Influence combination methods define the probability distribution of the random variables from probability information contained in multiple fragments which however also have a default distribution. A (possibly empty)

subset of the input variables are the hypothesis variables which represent human imposed knowledge.

Bayesian networks are assembled from the available network fragments for a specific situation. Network fragments are maintained and managed in a knowledge base, keeping track also of the object hierarchy and immediately propagating modifications to the related network fragments. The defined model construction process retrieves fragment classes from a knowledge base, creates fragment instances and combines them into the model workspace. The authors propose that search algorithms could involve computing or approximating bounds on the influence of a variable to decide whether the computation involved in extending the model is justified by the potential improvement in accuracy. D-separated variables would consequently be excluded.

The work in [258] controls the combination of fragments based on *data association* (deciding if a fragment's evidence is of a relevant domain), *hypothesis management* (generating and pruning hypotheses about domain entities and their interrelationships), and *pattern replication* (need for multiple copies of a network for temporal reasoning).

Figure 4.5 shows an example of several network fragments to be combined. Random variable D is a hypothesis variable, variables depicted with grey circles are input variables, and variables depicted with white circles are resident variables. The combination of the fragments shown on the left side produces the combined BN shown on the right.

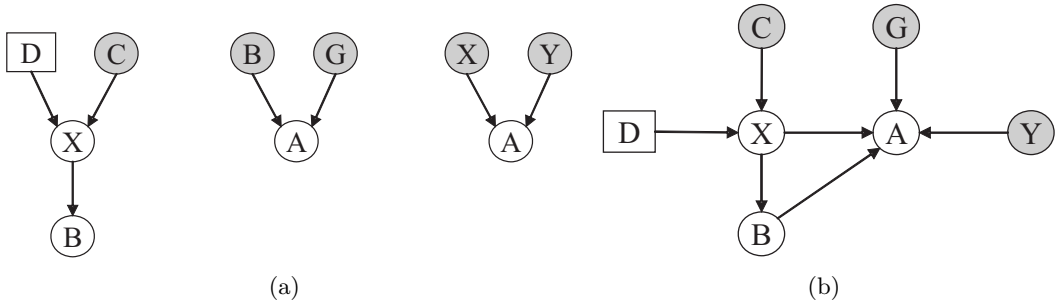


Figure 4.5: Networks fragments to be combined (left) and combination result (right).

Three methods to combine network fragments are proposed:

- *Simple-Combination*: This method is applicable if the combined network fragments contain only one resident variable which is not shared between network fragments. Their distribution is then defined purely by its parents within the own network fragment.
- *Default-Combination*: In the case the above assumptions do not hold, the default probability distribution is overridden by the resident variable's influence combination method.
- *Partial influence combination*: If a resident variable to be combined cannot access the information of all of its parents, partial influence models, e.g. independence of causal influence (ICI) models are used. The most widely used concept therefore is the *noisy-OR* [89, 66] operator.

Evaluation of Bayesian networks based on network fragments follows then a standard approach supporting the custom combination method. In the cases of Default-Combination and partial influence combination, inference yields only approximate results belonging to the class of *anytime model construction*, see section 2.3.3.3 on page 38. Distributed inference as necessary for resource constrained and mobile devices is not intended.

4.4.1.3 Object Oriented Bayesian Networks

Object Oriented Bayesian Networks (OOBN) is a proposal of Koller and Pfeffer [257] from 1997. They use concepts from object oriented programming for the specification process of BNs. An OOBN consists of a single situation object that is using several *Object Oriented Network Fragments* that define its probabilistic properties. OONFs are DAGs using the value attributes and having the the input variables as interface.

Also Bangsø defined an OOBN-Framework in 2000 [245]. Both frameworks are based on MSBNs (see section 4.4.1.1) and have roughly the same expressiveness using different vocabulary and slightly different structural implications which is why both approaches can be described in this section. Bangsø's framework goes a bit further however, providing means for top-down modelling and time slice representation [244] and also dedicated, enhanced inference methods [251, 246].

The core idea for OOBNs is the object oriented point of view and similar to Laskey's and Mahoney's idea with network fragments (see section 4.4.1.2): a model separated into several elements can be combined in different situation specific constellations without multiple definition of elements.

Each BN module is an object in OOBN and as such an instance of a class. Objects that instantiate the same class have the same attributes and structure. BN objects are seen as functions from input to output RVs, providing a probability distribution for the output RVs. They distinguish three different types of RVs: *input* variables are *basic or structured variables*, *output* variables and *encapsulated* variables so called *value attributes*, which are objects themselves. Figure 4.6 shows an example of an OOBN structure.

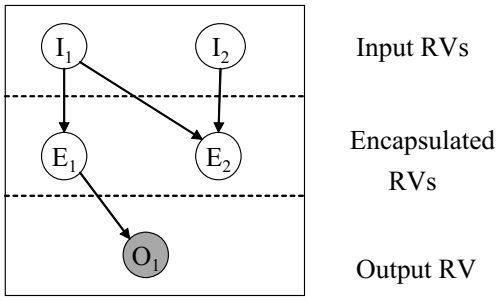


Figure 4.6: Subsets structured example object [245].

The marginal differences are that in Bangsø's approach input nodes can refer to nodes outside the object, while this is not possible in Koller's approach. Besides, in the latter all output nodes have to be used. Bangsø's approach allows for more flexibility, for instance by using prior probability distribution for uninstantiated input variables.

For inference in OOBNs, Koller presents the approach to transform the OOBN into a MSBN. Bangsø in addition considers the construction of a regular BN to allow for a greater variety of inference approaches or to directly construct a junction tree from the OOBN instance tree maintaining the structure of the OOBN and using then the PPTC algorithm.

To allow for frequent structure changes in highly dynamic domains, Bangsø introduces an incremental triangulation method [250] with which only modified parts of a junction tree have to be re-built. This method is based on the *Maximal Prime Subgraph Decomposition (MPSD)* [261].

A MPSD is derived from a junction tree by aggregating those pairs of cliques where the nodes in the common separator were not all interconnected in the moral graph. The

work of Olesen and Madsen in [261] has shown that only those parts of the BN are affected by structural changes, for which the MPSD has changed. Applying this technique a new junction tree can be compiled faster for a new situation, allowing then again to apply the PPTC algorithm for inference.

4.4.1.4 Pavlin's and de Oude's Modular Bayesian Networks

In parallel with the first publications of concepts of the research at hand, also Pavlin and de Oude have started publishing interesting concepts for modular Bayesian networks for information fusion, among others in [267, 265, 262, 264, 263]. An implementation of their concept of modular Bayesian networks is given in the *Distributed Perception Networks*. They represent a subkind of MSBNs (see section 4.4.1.1), sacrificing some of the modelling capabilities in exchange of the improved support for dynamically changing environments. For complex structures of modules, building junction trees for inference would be very time consuming and involve a lot of synchronisation in a distributed system. As significant delays are not acceptable in highly dynamic and therefore fast changing environments, it is advantageous to use simpler structures.

In comparison to MSBNs, restrictions are in particular imposed on the number of nodes shared between modules. Furthermore, the extra-modular parents of a node are considered independent and uniformly distributed. This is a severe restriction unless it is known that there will be either hard or soft evidence added to it. Modules are arranged in trees. If the modules would build an undirected cycle (i.e. violate the tree structure), they are artificially d-separated at one of the connecting nodes by assuming evidence in it. Also this is only possible without loss of exactness, if such nodes are known to be observed.

Although the authors target a highly dynamic system, all BN modules need knowledge about the whole BN. This is used first to distribute the local outcome posteriors to all connected modules and second to check the *global I-mapness* in the modular BN after the addition of a new module, i.e. whether the new joint modular BN still correctly represents to conditional independences of its full joint probability distribution.

The basic assumptions together with specific assembly rules presented in [265] allow for distributed inference in the modular BN. The focus of inference thereby only concentrates on diagnostic reasoning which avoids message passing in both directions, but limits inference to only the root hypothesis of the complete modular BN.

The distributed inference is possible, if there are no multiple connections between the BN modules, i.e. if they form a cluster tree. Via the factor graph [59] and the partition graph used for MSBNs [271], possibly existing cycles are detected and resolved by clustering the loop causing node with other nodes, or by instantiation of nodes that are assumed to be observed – a step which requires knowledge about the whole system and which is not possible in a distributed way, however.

If the modules form a tree, all modules can be inferred locally and independently from other modules, given their Markov boundary which always coincides with its interface nodes [265]. Every local, modular BN has therefore always the same structure and the corresponding junction tree can be computed prior to operation, saving time later on.

The assumptions of Pavlin's and de Oude's work do not hold in the general case of context awareness. Neither is it clear in general, for which information evidence will be available, nor should a local module have to know which other Bayeslets are using its information. Heterogeneity of data and therefore an integration of dynamic probabilistic information or modules not including Bayesian networks is not addressed in de Oude's and Pavlin's work.

As the preconditions for this approach, as for de Oude's and Pavlin's work in general, do not always hold for context aware computing, their approach may serve as an example in real conditions, but has to be adopted for the requirements in this thesis.

4.4.1.5 Hwang's Modular Bayesian Networks for Landmark detection

Hwang and Cho in their research [253, 254, 255] collect all information locally available from mobile smartphones, in order to detect high level context, labelled not very consistently by the authors as *landmarks*, such as *Emotion and status*, *Everyday life*, *Event*, and *School life*. To cope with real-world irregularities, like varying levels of attention and emotions, inaccuracy of sensors, and uncertain causal factors [255], Hwang and Cho use static Bayesian networks, however not causal networks, designed mainly with boolean random variables instead of multi-valued discrete RVs.

They show in [254] that the resulting monolithic BN with 115 nodes and 298 parent edges cannot be evaluated with PPTC [88] on a smartphone with MS Pocket PC 2003 with 44 MB RAM. To reduce the computational demands of inference, they manually divide the monolithic BN into modules determined by the landmarks. The K2 learning algorithm [71] for the monolithic BN is adapted to order nodes not only topologically, but also based on mutual information [105]. Like this it can find a close-to-optimal distribution of nodes across the predefined BN modules. All the observed evidence is regarded to be independent from each other.

Hwang and Cho use this approach successfully to reduce the computational demands [254]. However they do not use the more sparse, hence more efficient representation of causal networks [66]. This approach still requires much human involvement, for determining the different modules, but also for clustering location coordinates and to discretise the measurements into time slots.

The inference approach of Hwang and Cho from [253] is a two stage approach, where every module first infers the contained landmarks given the available hard evidence. A model simplification is that hard evidence is used in more than one module without modelling the implicit dependency. In a second stage, the outcome of landmarks in other modules that are connected as parents is introduced as soft evidence with a technique they call "virtual linking".

The authors do not disclose in detail how this method works. The way how they apply the soft evidence and the assumptions made regarding the network structure (e.g. trees or polytrees) however determines, whether their approach infers exact posteriors or only approximations. From the available publications with some foreshadowed model simplifications an approximative solution can be assumed. In any case only deductive reasoning is possible, reversal of the inference direction is not.

In Hwang's and Cho's approach, only local information is taken into account, and together with the human involvement this makes clear that the approach is a fairly static one, not appropriate for context inference in a very dynamic environment. Ubiquitous computing however is very dynamic where nodes cannot be assumed to be always observed, not all other modules are known and information is distributed among different devices.

4.4.1.6 Hierarchical and Hybrid Bayesian Networks

An interesting approach is the work of Tu et al. in [270]. It proposes an architecture for identifying terrorist threats with Bayesian techniques. As intelligence agencies however are usually spread across different locations, a distributed model is proposed. It can

incorporate not only static Bayesian networks, but also HMMs (see section 2.3.5) in a two-layered approach, as represented by Figure 4.7.

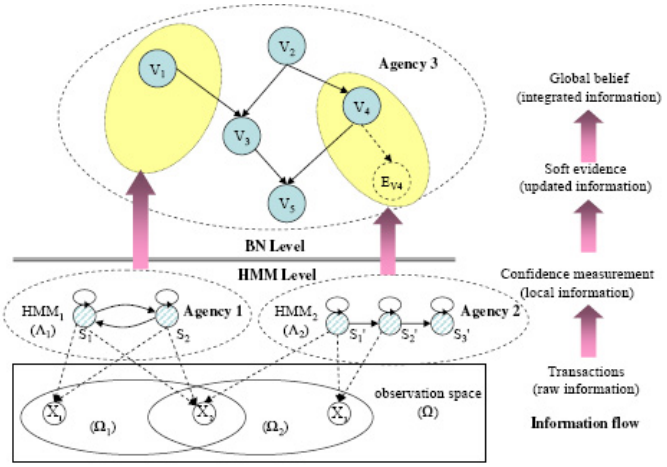


Figure 4.7: Example of a Hierarchical and Hybrid Bayesian Network consisting of two HMMs Λ_1 and Λ_2 on the lower layer and a static Bayesian network on the top layer [270].

The HMMs situated on the lower layer represent sub-models that are populated by domain experts with evidence. On the top layer a static BN represents the overall situation, in the given example at a "higher level agency" which is entitled to take decisions. It incorporates the available (hard) evidence as well as the outcome of the HMMs in their representation nodes by soft evidence and evaluates this BN to infer the current level of threat.

This model allows only for inference in one direction from the lower to the top layer and does not consider efficiency augmentation by distributing the inference of the static BN. It represents a static model of a situation, designed by a domain expert and only manually changeable. Inference between different modules in one layer is not foreseen, just as little as more than two hierarchy layers for complexity reasons (in modelling and inference) [270]. Hence, although the idea is promising, in this form it is not applicable to the conditions of ubiquitous, context aware computing.

4.4.2 Discussion

The inference approaches in split Bayesian networks can be grouped into three categories:

1. Different modules are composed to a common structure for inference.

2. A complete BN is used to set up a separated inference structure in which independent and separate inference is possible.

3. Other approaches realise completely independent and distributed inference.

Examples for the first category with a common inference structure are the network fragments of Laskey and Mahoney [258] or also Paskin's and Guestrin's approach to modular Bayesian networks in [266]. Also OOBNs offer this option with which then standard PPTC for inference is possible. With this approach a BN can be adapted to the needs of a specific situation and under certain conditions reduced in size. Only in this case, inference duration is reduced. Therefore, in general no significant inference improvement can be obtained. Moreover, as the combination of all knowledge on one computer is necessary

for inference, privacy is not protected by this approach. Hence this category does not represent an appropriate choice for context inference on resource constrained devices.

The second category is used in Xiang's MSBNs [271, 273, 272], and consequently also in the OOBN approaches and Pavlin's and de Oude's modular BNs which build upon MSBNs, but also by Hwang in [254]. Inference duration is improved by building a common secondary inference structure of the different modules (like a junction tree) in which different modules can be evaluated in parallel. However it does not make the different modules independent from each other and the system has to have knowledge about all existing connections between the modules. The dynamics of a system determine the usefulness of such approaches, as the construction of the secondary structure is more expensive than for normal BNs. As Ubiquitous Computing environments are highly dynamic and the large scale does not permit overall knowledge about all possible inference connection, neither this category is suitable for Bayesian context inference.

The requirements of large scale, highly dynamic environments are fulfilled in general by the third category, which however restricts the inference approaches to provide approximations of the searched posterior probabilities. The inference approaches of Hwang's modular BNs or the hierarchical and hybrid BNs could realise this, but are not ideal implementations of this idea. Both approaches only allow for static modelling and do not optimise the network and dependency design which is necessary to optimise inference time. Both approaches use knowledge of the overall BN to distribute information into different modules in the design phase. Dynamic construction of networks of BN modules is not considered.

Only Tu's work [270] allows for the integration of different inference approaches. This however would be a requirement for context inference in ubiquitous computing systems where a plethora of different inference methods will exist.

The aim of the present research hence is to find a viable combination of the advantages of all existing systems. It has to allow for tailoring a BN to situation-specific needs like network fragments and OOBNs by a comprehensive knowledge base of BN modules, well-defined interfaces, as well as efficient dynamic selection and combination of relevant modules for completely independent inference. The privacy protection facilities of the distributed inference approaches has to be maintained, and the inclusion of different inference methods has to be offered.

Chapter 5

From Bayesian Inference to Context Inference

This work proposes Bayesian techniques for context inference. Their theoretical bases have been presented in section 2.3, existing work for context inference and extensions to the Bayesian theory improving its tractability in chapter 4. The following chapter presents the concept for the adaptation of Bayesian techniques to cope with the requirements of tractable and personalised context inference in ubiquitous computing on mobile and resource constrained devices.

Therefore the first section will show how to integrate Bayesian context inference with context management systems in order to reduce resource consumption. Then, in sections 5.2 and 5.3 the inference of lower level context is described with the examples of location and human motion patterns. These are input to high level context inference in a fully Bayesian, hierarchical inference architecture.

Once the reader has seen how lower level context is inferred, the next step is the identification of those contextual aspects that are most relevant for a given inference goal. Therefore section 5.4 will present an approach for tractable context inference with modular Bayesian networks, show how modular inference structures are minimised, assembled, and efficiently evaluated.

5.1 Applying Bayesian Techniques in Context Inference

The requirements presented in section 3.3 on page 58 have two-fold consequences: first, they require a model of context inference rules (see Definition 10) which is modular, allows for heterogeneity and is efficient to be inferred upon. Second, the context management system has to support creation, selection and assembly of the inference rules, as well as to control the inference requests in order to allow for short response times, which is critical given the NP-hardness of probabilistic inference as shown in 2.3.3 on page 34.

While the exact properties of the modular, locally evaluated Bayesian context inference rules will be described in section 5.4 from page 129 onwards, this section deals with the implications of context inference to a context management system (CMS). Requirement (5), "Access control and privacy" for inferred context, is neglected, as with local and modular inference it is not related specifically to context inference any more and can be treated like general access control to context information, such as in [11].

This sections deals in particular with the life cycle of context inference modules and triggering of inference in a context management system, adopting the ideas presented by the author et al. in [118].

5.1.1 Creation, Storage and Access of Context Inference Rules

One challenge for CMSs is the handling of context inference rules. When and how they are created, stored and accessed are important parameters to guarantee short response times of a CMS. The focus lies on Bayesian Networks as inference rules, although in general all classification models can be used, as long as there is a suitable evaluation algorithm and the results are compatible with those of Bayesian context inference rules.

5.1.1.1 Rule Creation

This work promotes the usage of Bayesian networks as context inference rules. If the BNs are modelled correctly, context inference can be performed by probabilistic inference as described in section 2.2.5 on page 25. Different from e.g. Gu and Zhang in [124], this work models context types (e.g. *PhysicalActivity*) as random variables.

The different values a context type can adopt (e.g. *running*) define the RV's value range. Modelling has to take care that the value ranges are always exhaustive and that the values are mutually exclusive. A set of standardised but extensible ontologies as they were proposed by Strang, Linnhoff-Popien and the author in [42] will assist rule creation among different domains. Exhaustiveness can always be guaranteed by an additional *other* state.

The random variables defining the Bayesian context inference rule shall be connected in causal ordering, allowing therefore to represent the domain as sparse and efficient as possible [66].

Rule Creation can be triggered either on demand, i.e. when inference has been requested but there is no existing inference rule for the requested context type, or by using independent mechanisms, e.g. a nightly running batch mode. There are different approaches for the creation of Bayesian networks for context inference:

- "manual" creation by a domain expert,
- adaptation of predefined templates for an individual, optionally by the user himself,
- service providers can share context inference rules for context specific to their service,
- automatic learning from the history of context with the techniques described in section 2.3.4 on page 39.

The information used for automatic learning of Bayesian networks, the context history, is an important factor for learning. Usual learning methods like the one described in [71] learn relations among all elements of a complete context history. Hence, for learning a rule about a high level context attribute, values about this attribute have to be known and present in the history together with other context information before the relations found between the different context attributes can be modelled in a rule. This implies that the high level context originally has to come from human interactions with the context history. To enable data collection among different users, a common understanding of existing context attributes has to exist, which encompasses the existence of a common context ontology (cf. Definition 11 on page 17) defining the existing context types.

Moreover, algorithms for learning BNs from incomplete data sets (algorithms from the class of (structural) expectation maximization (EM) introduced in [74], cf. section 2.3.4.3) will not be able to give semantically meaningful names to newly identified context nodes. Hence, they cannot fully overcome the need for human interaction for context definition and labelling, as context attributes without semantically defined name cannot be queried by external services.

Because of the large amount of information logged in the context history, it is necessary to restrict the scope of learning as far as possible, e.g. by incorporating knowledge from other already existing rules from different users, by individualising templates created by human experts, or by limitations specified by domain experts or the user.

Limiting the amount of involved context types, imposing known causal dependencies and specifying conditional independencies constrain the search space of structure learning and help to find a structure ideally representing the past evidence.

Just like for inference rule creation, the update of existing rules can also be based on regular processes or by incremental incorporation of new knowledge. The decision has to take into account the system's response time, but also the quality and up-to-dateness of the responses.

A CMS has to provide all rule creation and update processes to fulfil requirement (6) from section 3.3. On-demand rule creation adds significant delay on a response, given that learning (even if only coarse grained) takes some seconds, but is necessary as delay may still be better than giving no response at all. It can remain for the requester to decide if it continues after some seconds or disregards the answers and proceeds immediately without that information. Incremental rule updating keeps all existing rules always up to date at the cost of permanent computation load in the background.

Incremental rule updating however would only update a BN's parameter and not its structure. New rules or new structures cannot be provided incrementally, which however would be necessary for context awareness. So even if they are hardly flexible, resource intensive and not need-oriented, regular batch processes for learning rules will be necessary to discover new rules. To reduce the computational burden, these processes have to run at typical low-usage times of the systems, for instance during the night.

5.1.1.2 Rule Storage and Access

Context inference rules (CIR) have been postulated to be modular and to refer to local data only in requirement (1) of section 3.3. Local storage of CIRs bears the following advantages:

- individualisation to the device user's preferences,
- individualisation to the locally available information,
- reduced network traffic for remote information,
- easier privacy protection.

Rules can be stored and managed by the local CMS or be dynamically integrated as third party services in a CIR service registry upon availability. The latter approach is more generic, but reduces control over execution and update of the context management system, in particular updating and learning upon request.

Hence the approach in this thesis manages Bayesian inference rules inside the context management system. The resulting integration of CIRs in a CMS with its life cycle is shown in Figure 5.1.

When a user owns several mobile devices, synchronisation mechanisms for distributed context management systems have to be used to maintain a complete and consistent state on all devices. The smart space concept can solve this by installing proxies to remote inference rules, just like for access other users' context [134]. When a locally evaluated Bayesian network refers to a random variable for which there is no local evidence, it

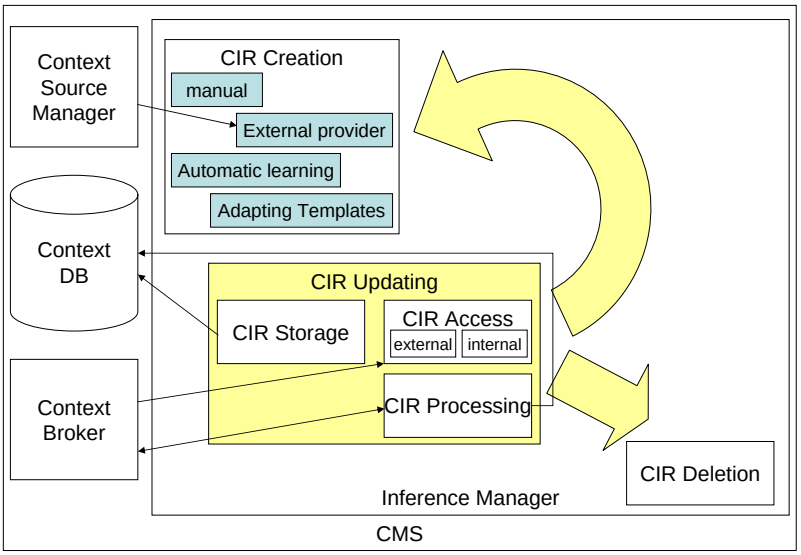


Figure 5.1: Lifecycle of context inference rules in a context management system.

requests inference rules or proxies in the local CIR storage system and requests the remote evidence via the respective proxy connection. If access is granted, the remote evidence can be incorporated.

5.1.2 Inference Scheduling

Once a solution about the maintenance of the inference rules has been found, it has to be assured that the inference process itself can run as fast as possible. Therefore it has to be scheduled efficiently: inferred context information has to be sufficiently current, but unnecessary inference, hence resource consumption, has to be avoided.

5.1.2.1 Triggering Probabilistic Inference

Context inference should never be directly called by a consumer. It should be transparent, whether requested information has to be inferred at all or whether it is provided by a context source. Consequently the CMS is receiving all requests in the *context broker* and forwards them to the context inference system if necessary, i.e. if no appropriate and up to date information is available in the CMS.

The easiest way to always have the current information stored in the database implies permanent evaluation of the inference rules, but this results in unnecessary evaluations and many storage processes which will not have been used before they are overwritten.

The other extreme is to initiate inference only on request, as shown in Figure 5.2. This is problematic, as response time increases and, more important, it would eliminate the option for context consumers to register for changes in specific high level context. Hence, preferences, proactive service invocation rules or other inference rules could not be based on inferred high level context. In the scenarios of the introduction chapter on pp. 3 et seqq., it can however be seen that this is the way how daemon services running in the background incorporate context in order to proactively start actions when appropriate. To enable such context subscription, continuous inference based on the update of the requested rules input nodes has to be offered.

In order to avoid unnecessary evaluation, a hybrid solution will perform best. It should stick to on-demand inference as the default case, but enable subscriptions with a

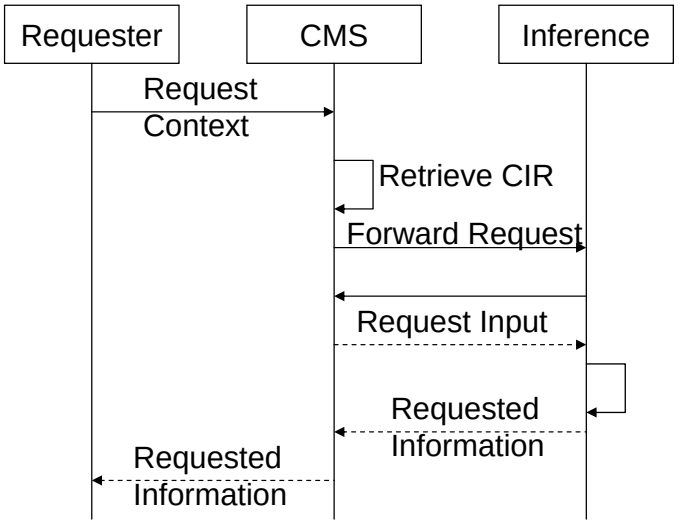


Figure 5.2: Message Sequence Chart: context inference on demand.

bypass: No rule should be evaluated that was not explicitly requested. When a consumer subscribes for a context attribute which has to be inferred, the respective rule should be evaluated and the CMS internally subscribes this rule as a consumer of all its input context. This process is shown in the message sequence chart of Figure 5.3. Like this, inference is triggered on necessity, i.e. depending on its input information, instead of a only temporally regular basis.

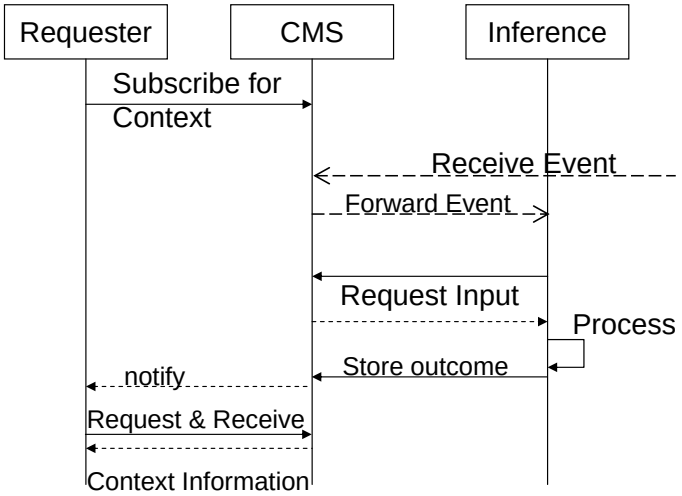


Figure 5.3: Message Sequence Chart: continuous context inference.

5.1.2.2 Update Frequency

The compromise solution could still cause (too) frequent inference, e.g. because of a high frequency of an input context source. Inference frequency $f_{\text{inference}}$ for an inference rule that depends on the update frequency of its input information input_i, f_{input_i} can be defined as follows:

$$\max_i(f_{\text{input}_i}) \leq f_{\text{inference}} \leq \sum_i f_{\text{input}_i} \tag{5.1}$$

If there is one input information coming from a sensor with frequent measurements (e.g. an IMU provides measurements in the order of 100 Hz) this would cause so frequent rule evaluation that storage and inference would be only allowed to take few milliseconds. If the relevancy of this inference is not too high, this would be a waste of resources. In contrast, there are also cases of inference, e.g. the estimation of hazardous situations in *CACC*, where the prediction should never neglect input from its sensors.

The latter case however shall be neglected. It is unlikely that sensors would store data with update rates higher than 300 Hz in a CMS for cost and resource consumption reasons. It can be assumed that safety critical applications like *CACC* bring along their own sensors where they can directly access the raw measurements. Information that would be shared in a context management system would already be preprocessed and represent more meaningful and less volatile information, for instance *cornering* instead of the respective accelerations and turn rates.

Hence, in order to avoid too frequent inference, a CMS can forward context requests to the inference engine only when the quality of the available information is not sufficient. Measures for the quality are among others the uncertainty and the recentness of the information. If the context attribute hence has just been inferred and there is a state that has a significantly higher probability than other states, then also a reoccurring change of the input context would not trigger inference anew. With an appropriate threshold for *up-to-dateness*, always up-to-date context information can be guaranteed.

Taking into account that the necessary update frequency depends on the type of the context information and its use case, CMS cannot use always the same threshold for rating information as up-to-date or outdated. Expert knowledge or human made ontologies will have to provide this information and be combined with a CMS-immanent minimal boundary between two inference calls. A reasonable boundary would be 1 Hz, as argued in section 3.2 on page 57 – the approximate frequency of a human step.

5.1.3 Summary

This section has shown how to use modular Bayesian networks as context inference rules. Rules can be provided to a CMS manually by the smart space owner or by external entities. The CMS manages forwarding of context requests to the rule evaluation engine upon need.

Automatic learning methods can assist the rule creation, but need human involvement in terms of labelling data and restricting the search space for structure learning. Automatic creation and updating of inference rules needs both, resource intensive batch algorithms which can incorporate new knowledge into the BN structure, and light-weight incremental algorithms which can react faster on changed behaviour and adapt the parameters accordingly.

Inference rules should be stored and used only locally in the own smart space. The frequency of inference has to be controlled by the CMS, allowing for context subscriptions, but using on-demand context inference whenever possible. Inference frequency for subscribed context should be limited by both, update frequency of low level input context and a context attribute dependent minimum update distance.

5.2 Handling Location Information in Context Inference

The probably most important and most used context information is location. An important criterion for the practical usefulness of the approach proposed in this thesis is its applicability to location – absolute location, the outcome of precise positioning, as well as symbolic locations which are of high importance for inference or extraction of repeated patterns in a user's behaviour. Hence this section designs inference modules for location inference which can be used upon need for high level context inference.

After discussing the relevancy of location for context aware systems in section 5.2.1, an example for inferring precise position with Bayesian methods is given in section 5.2.2. In section 5.2.3 a way to transfer absolute to symbolic location maintaining the uncertainty information is presented, before section 5.2.4 shows how to model this information in static Bayesian networks for context inference.

A derivation of absolute location is *proximity*, a further important factor for the inference of high level context. Section 5.2.5 presents a concept for its inference from absolute location and its integration in Bayesian networks for context inference.

5.2.1 Importance

Albrecht Schmidt titled his paper [185] back in 1999: "There is more to context than location" – the existence of this title already indicates the outstanding role of location which has not changed in the last twelve years. Hardly any area has received as much attention as positioning (outdoors and indoors) and location based services.

Location based services are the first and most important instance of context aware services and the first ones that have reached a significant penetration of the market with application for mobile phones, inclusion in social networks and in particular navigation devices. A lot of research work has lead to appropriate algorithms and architectures, such as [177], which has made this success happen.

Location is also used in all example scenarios of section 1.2 in the introduction. The three scenarios presented in total 35 context aware services, of which 29 are impacted directly by the user's location. This is understandable, as human users are physical entities and as such always have a location. This location defines the set of possible interactions, of available services and environmental conditions – hence the user's situation.

The next generation of context aware services, including more and more information and sensor measurements [185], will undoubtedly still rely on location as one of the most important sources of context. In terms of context modelling, location is one of many attributes of an entity's context [19], describing in the terms of Strang [42] one aspect of context.

Fundamental for location dependent context aware services is precise positioning. In particular in indoor environments a deviation of only one metre (about the length of a step) can be decisive. It determines if a user is on one side of a wall or on the other.

Pervasive usage of location sensors requires efficient economising on resources. As such, information updates should only be considered when they are significant, i.e. the new location has a different semantic meaning than the last one. In the introductory scenarios it can be observed that, with 17 out of 29, a narrow majority of the context aware services are not depending on absolute position, but on symbolic location like rooms, train stations or public places.

Therefore, precise location of a user not only has to be translated into discrete, semantic positions, but also has to be available in different levels of detail. Only with very high precisions (e.g. different rooms in a building) demands of certain applications can be

fulfilled, but in order to save resources, less details should be taken into account in areas less relevant for inference.

5.2.2 Fusion of Location Information

This work does not focus on location estimation, but shows a viable approach to integrate it into high level context inference. Therefore a slim inference module is presented here that shows how to infer precise position and can be integrated in high level context inference.

As an example an indoor environment is discussed where no satellite navigation is possible and other location sources have to be fused for inference of precise position. The proposed approach uses Wi-Fi fingerprinting and an INS as information sources, neglecting other infrastructure based information like maps. The description follows the work of the author et al. from 2008, see [151].

The goal is to obtain and process all the sensor data locally, and *without any need of registration with the local infrastructure.* WLAN fingerprinting based on the signal power (e.g. [176]) shall be used for the long-term stability of the approach, against the drift of the inertial system. The only information needed at the local device will therefore be a fingerprinting database for the current building, which can be maintained and distributed by an entity independent of the local wireless infrastructure domain. As few calibration locations as possible should be used, relying between these on the short-term accuracy of foot mounted inertial dead-reckoning (for instance ZUPT based techniques, [150]).

This approach is novel by relying on the INS with ZUPT for improving Wi-Fi fingerprinting, neglecting the building layout and using an efficent, hierarchical, close to optimal Bayesian inference approach. Furthermore, in this work a magnetometer is used for estimating the orientation of the user.

The work of Woodman et al. [193] uses very coarse WLAN positioning to reduce the initial ambiguities of map aided inertial navigation. The work in [186] describes how fingerprinting can be simplified by using an INS (not foot mounted) during calibration and how actual performance is enhanced during positioning. An approach using a foot mounted INS will perform better in situations where WLAN positioning is not available for any significant length of time during which a standard INS approach (no foot mounting; no ZUPT) would drift too far. This also applies to the work of Evennou in [149] in which no true 6DOF ZUPT based inertial processing was performed, but only stride estimation (angular change and stride detection). Furthermore, the particle filter used in [149] relies on a known building layout.

The remainder of this section will first present the approach in general, describe the information sources *Wi-Fi fingerprinting* and *Step Detection with an INS*, before it presents the filter for the fusion of both information sources.

5.2.2.1 Location Estimation Approach

Prerequisite for an inference module for location is its autonomous evaluation, independence from infrastructure, and therefore also computational efficiency, in order not to overburden resource constrained mobile devices. Techniques where the location is partially or fully computed in the infrastructure will always remain questionable in terms of privacy considerations. Suitable (approximate) Bayesian sensor fusion algorithms provide a close-to-optimal estimate of the position that can be efficiently implemented on the end-user's device.

The approach presented here requires no processing outside of the local device and minimal a-priori fingerprinting effort. A hierarchical Bayesian filtering approach using

cascaded extended Kalman filters (see section 2.3.6.2) to achieve a real-time implementation is employed. A schema for this approach is given in Figure 5.4.

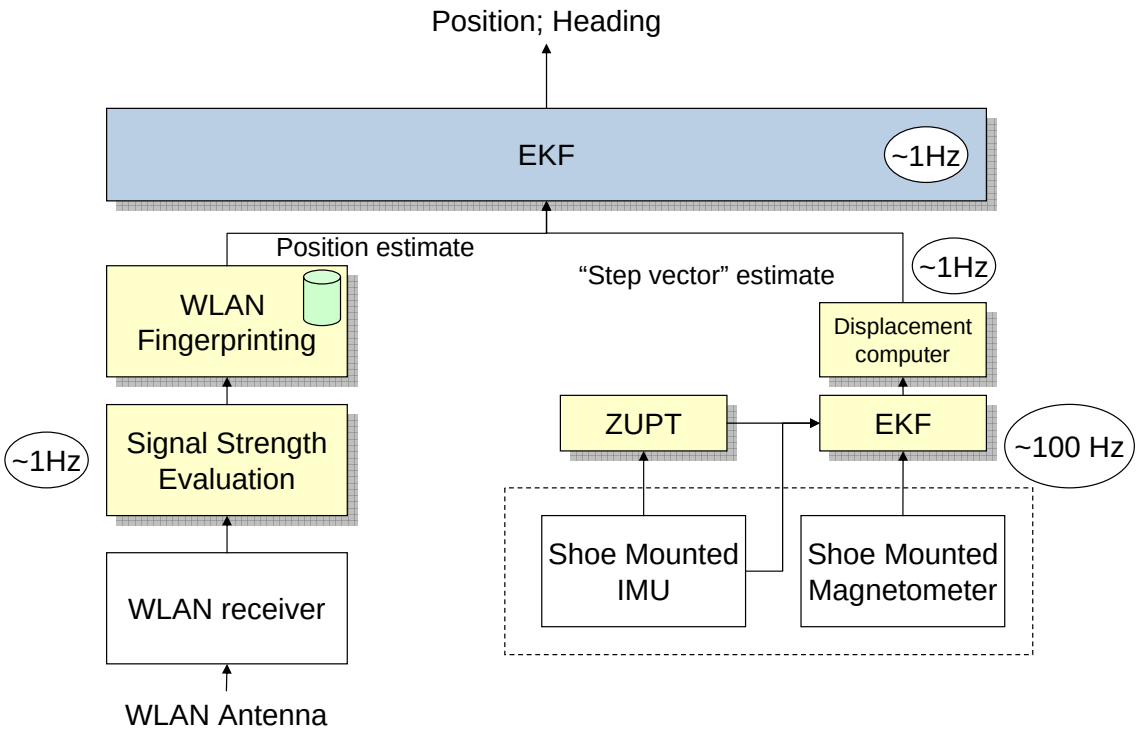


Figure 5.4: This figure shows the complete system with two layers of processing: a lower one for the WLAN position estimate and the step computation which are then fused in an upper EKF.

This hierarchical approach has got a separate Bayesian filter for the inertial system which estimates individual steps of the user; these estimates are then combined with the estimate of the location from fingerprinting. Such decoupling allows the estimation filters to run at their local sampling rates and reduces overall complexity without suffering from significant loss of final estimation accuracy [169].

Using the (extended) Kalman Filters fulfils the requirement of resource efficiency in terms of computation: while providing an (almost) optimal solution, the computation is extremely fast, as it relies only on matrix operations.

The use of Wi-Fi fingerprinting fulfils the requirement of needing no association with the actual access points and is in combination with the INS also relatively resource efficient in terms of energy consumption. Scheduling of the WLAN receiver's duty cycle can be controlled by the INS: the user's motion can be detected from the foot mounted sensors which can trigger the WLAN module to scan the signal strengths of access points, when the user starts moving, as only in this case the signal strengths are relevant.

5.2.2.2 Wi-Fi Fingerprinting

The method used for Wi-Fi fingerprinting in this work is a variant of location fingerprinting using probabilistic algorithms (compare to section 4.2.1.2 in the description of the state of the art), modelling the location calibration points with discrete RSSI probability distributions (see Figure 6.2 on page 159 for an example). The system used has two phases of operation, a calibration phase, and a location calculation phase, which can be initiated after calibration.

During the calibration phase a database of location fingerprints is established using RSSI measurements at a number of calibration points, each referenced to a physical location. The functional components of the system can be seen in Figure 5.5. The RSSI Manager collects readings from the WLAN driver, and processes this information to establish a fingerprint, which is then stored into the fingerprint database. This process is then repeated until calibration points have been set for the entire area of interest.

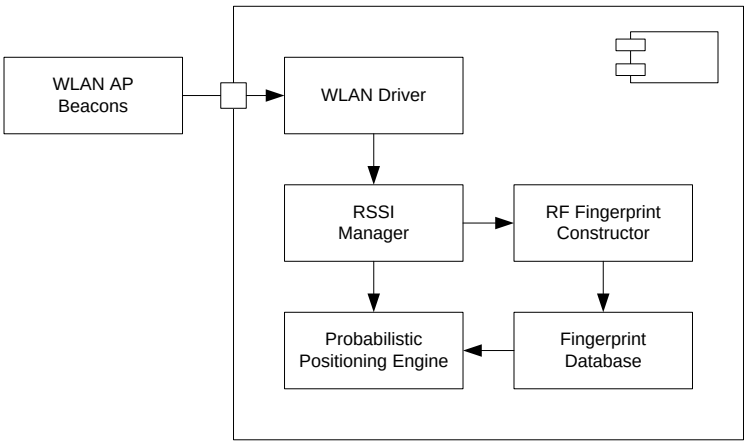


Figure 5.5: Functional components of a probabilistic Wi-Fi fingerprinting based indoor positioning system.

The more calibration points are used, the greater is the accuracy of the system. However if calibration points are placed too close together, such that there is little difference between RSSI variation of access points, i.e. they have very similar fingerprints, little advantage is gained, and training time becomes infeasible for large areas. When fingerprinting is fused with an INS, the system needs less calibration points, as the accuracy between them can be ensured by the INS.

In the positioning phase, continual scans from the wireless driver are processed by the RSSI manager (see Figure 5.5). These readings are passed directly through to the positioning engine, which calculates a distance to each calibration point, comparing each one to the observed readings. The distance for each calibration point c is defined by the following equation:

$$\mathbf{D_c} = \sum_{\mathbf{i}} \sum_{\mathbf{j}} |\mathbf{x_{ir}} - \mathbf{x_{ij}}| \, \mathbf{p_{ij}} \ , \tag{5.2}$$

where x_{ir} is the received RSSI (in dB) for the access point i, j are the sample RSSIs defining the probability distribution for each access point, x_{ij} is the recorded RSSI stored in the fingerprinting database, and p_{ij} is the probability of measuring the reading at the given calibration point. $\sum_j |x_{ir} - x_{ij}| \, p_{ij}$ is hence the expected distance to access point i. Summing over all access points, then gives a distance calculation for each calibration point denoted $\mathbf{D_c}$. The results for all calibration points are collated, and the minimum distance is taken as the location estimation.

To increase stability of the given location, a best-of-three approach is used to establish the estimated location. If no dominant location is determined, the latest estimate is returned as the current location.

5.2.2.3 Step Detection

This work uses a foot mounted Inertial Navigation System and determines the wearer's position in a PDR approach with ZUPT (cf. section 4.2.1.2 in the state of the art description). In the following the details of the inertial PDR filter are addressed.

Algorithm Fundamentals:
A strapdown navigation algorithm [188] processes the vector of acceleration and turn rate measurements $\mathbf{z}_l = [\mathbf{a}_l \ \boldsymbol{\omega}_l]^T$ provided by the inertial sensors to compute position $\mathbf{R}_l$, velocity $\mathbf{V}_l$, and attitude $\boldsymbol{\Psi}_{\boldsymbol{l}}$, i.e. the heading of the shoe wearer. In parallel an EKF is used to estimate the errors of the strapdown calculations [155].

In this work, nine states are estimated by the filter: position errors $\delta\mathbf{R}_l$, velocity errors $\delta\mathbf{V}_l$ and attitude errors $\delta\boldsymbol{\Psi}_l$. A magnetometer is incorporated to align and stabilise the heading as an additional measurement. Hence the inertial filter provides estimates of position, velocity, and attitude in terms of a Gaussian PDF. In the subsequent processing only position and heading are states of interest:

$$\mathbf{X}_l = \begin{pmatrix} \mathbf{R}_l \\ \Psi_l \end{pmatrix} \ , \tag{5.3}$$

where Ψ_l is the yaw angle derived from $\boldsymbol{\Psi}_{\boldsymbol{l}}$. From the posterior PDF of the inertial filter the (marginalised) posterior $P(\mathbf{X}_l|\mathbf{z}_l, \ldots, \mathbf{z}_0)$ can be derived straightforward.

Rest Phase Detection:
The reliable identification of the foot's rest phases is crucial for the update of the PDR filter. Different approaches have been proposed to trigger the ZUPT measurement [150], [153]. This work basically follows these ideas and monitors the magnitude of the acceleration vector, which is sensed by the accelerometer triad. If the signal remains within a threshold interval around earth acceleration for a certain time interval ZUPTs are triggered until the threshold condition is violated.

Step Sensor:
An extended Kalman filter is used to process the high rate inertial measurements, further on called *inertial filter*. To exploit them in the upper layer fusion filter a (virtual) step sensor is derived from the output of the inertial filter, which provides a measure of the travelled distance and the change in heading for each step of the pedestrian, see Figure 5.6.

Each time a new ZUPT is triggered the expectation of the inertial filter $\hat{\mathbf{X}}_l$ is stored in the variable $\hat{\mathbf{X}}_L$. Introducing the step displacement variable $\Delta\mathbf{X}_L = \mathbf{X}_L - \mathbf{X}_{L-1}$, the displacement with respect to the coordinate system of the inertial filter can be written as:

$$\begin{aligned} \Delta\hat{\mathbf{X}}_L &= \hat{\mathbf{X}}_L - \hat{\mathbf{X}}_{L-1} \\ &= \begin{pmatrix} \Delta\hat{\mathbf{R}}_L \\ \Delta\hat{\Psi}_L \end{pmatrix} \ . \end{aligned} \tag{5.4}$$

The displacement with respect to the last heading (before the step) is computed:

$$\Delta\mathbf{x}_k = \begin{pmatrix} \mathbf{C}^T(\Psi_\varepsilon)\mathbf{C}^T(\Psi_{L-1})\Delta\hat{\mathbf{r}}_L \\ \Delta\hat{\Psi}_L \end{pmatrix} \ , \tag{5.5}$$

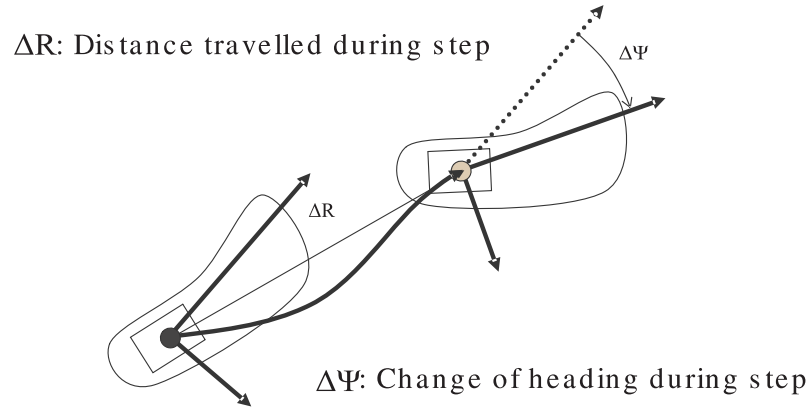


Figure 5.6: The INS is used to calculate an estimate of the pedestrian's step in the form of a foot displacement vector.

with the rotation matrix $C(\alpha)$ for a rotation by angle α:

$$\mathbf{C}(\alpha) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} . \tag{5.6}$$

The average heading misalignment of the inertial sensor platform with respect to the pedestrian's heading is given by the angle Ψ_ε, which has to be fixed initially.

5.2.2.4 Joint Sensor Data Fusion

As the next step, the WLAN fingerprinting measurements have to be fused with the detected steps of the INS. This is done via the upper layer filter in Figure 5.5 in which the pedestrians position $\mathbf{R}_k$ and heading Ψ_k at time k are tracked. The state vector can thus be written as:

$$\mathbf{X}_k = \begin{pmatrix} \mathbf{R}_k \\ \Psi_k \end{pmatrix} . \tag{5.7}$$

To ease the incorporation of the step sensor only the change in position $\Delta\mathbf{R}_k$ and heading $\Delta\Psi_k$ is considered for each step through the step measure

$$\Delta\mathbf{X}_k = \begin{pmatrix} \Delta\mathbf{R}_k \\ \Delta\Psi_k \end{pmatrix} . \tag{5.8}$$

As defined in section 2.3.6 on page 46 et seqq., the filter is defined by a transition model that describes the evolution of the state variables and an observation model describing the relation between the observed variables and the causing state variables. The next paragraphs describe both models.

Transition Model:

The transition (or *movement*) model characterises the temporal evolution of $\mathbf{X}_k$ in order to reflect the physical constraints that are imposed on the movement of a pedestrian. In particular in an indoor environment this may include also any restrictions which are imposed by the building layout. The benefit of this approach was shown for example in [168] and [192]. For the reasons described above, the building layout is neglected however in this approach. Hence formally, the new value $\mathbf{x}_k$ of $\mathbf{X}_k$ is assumed to depend only on the previous state $\mathbf{x}_{k-1}$, the current step measure $\Delta\mathbf{x}_k$ and a noise term $\mathbf{n}_s$:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \Delta\mathbf{x}_k, \mathbf{n}_s) , \tag{5.9}$$

where we have chosen that the new location value $\mathbf{r}_k$ of $\mathbf{R}_k$ and heading ψ_k of Ψ_k depend on the past state and on the step-measure through

$$\mathbf{r}_k = \mathbf{r}_{k-1} + \mathbf{C}(\psi_{k-1})\Delta\mathbf{r}_k + \mathbf{n}_{s,r} \ , \tag{5.10}$$

$$\psi_k = \psi_{k-1} + \Delta\psi_k + n_{s,\Psi} \ , \tag{5.11}$$

where $\mathbf{C}(\psi_{k-1})$ is the rotation matrix given in Eq. (5.6).

The noise processes $\mathbf{n}_s = [\mathbf{n}_{s,r}^T, n_{s,\Psi}]^T$ and $\mathbf{n}_{s,r} = [n_{s,x}, n_{s,y}]^T$ are zero-mean uncorrelated Gaussian noise processes of variance $\sigma_{s,x}^2$, $\sigma_{s,y}^2$, and $\sigma_{s,\Psi}^2$ respectively which are adjusted to reflect the uncertainty of the step-measure.

Observation Model:

The position estimate obtained by Wi-Fi fingerprinting is used as an position measurement $\mathbf{z}_k$, an instantiation of the random variables $\mathbf{Z}_k$, in the main integration filter and is assumed to depend only on the current state $\mathbf{x}_k$ and a noise term $\mathbf{n}_w$:

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{n}_w) \ . \tag{5.12}$$

In particular we assume

$$\mathbf{z}_k = \mathbf{r}_k + \mathbf{n}_w \ , \tag{5.13}$$

with $\mathbf{n}_w = [n_{w,x}, n_{w,y}]^T$ being zero-mean uncorrelated Gaussian noise. The respective variances $\sigma_{w,x}^2$ and $\sigma_{w,y}^2$ are adjusted to reflect the uncertainty of the Wi-Fi fingerprinting based position estimate.

Filter Design:

Since the building layout or further sensors are not used, there is no need to incorporate any further non-linear constraints than the one given by Eq. (5.10). But this relation is rather moderate with respect to non-linearity and thus an extended Kalman filter [60] is adequate to implement also the integration filter, in particular as all relevant noise sources are Gaussian.

The implementation of the EKF follows strictly the process described in 2.3.6.2: Given the initial mean $\mathbf{x}_0$ and the associated initial covariance $\mathbf{\Sigma}_0$ each filter iteration in the *prediction step* recursively calculates the parameters of the Gaussian prior PDF, which are mean

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \Delta\mathbf{x}_k, \mathbf{0}) \ , \tag{5.14}$$

and covariance

$$\mathbf{\Sigma}_k^- = \mathbf{F}\mathbf{\Sigma}_{k-1}\mathbf{F}^T + \mathbf{\Sigma}_x \ , \tag{5.15}$$

where the Jacobian matrix [49, 60] of the system dynamics is given by

$$\begin{aligned} \mathbf{F} &= \left.\frac{\partial f(\mathbf{x}_{k-1}, \Delta\mathbf{x}_k, \mathbf{0})}{\partial \mathbf{x}_{k-1}}\right|_{\mathbf{x}_{k-1}=\hat{\mathbf{x}}_{k-1}} \\ &= \begin{pmatrix} 1 & 0 & g_1 \\ 0 & 1 & g_2 \\ 0 & 0 & 1 \end{pmatrix} \ . \end{aligned} \tag{5.16}$$

The terms g_1 and g_2 are the respective elements of the vector

$$\mathbf{g} = \mathbf{C}'(\Psi_{k-1})\Delta\mathbf{r}_k \ , \tag{5.17}$$

where the derivative of the rotation matrix is

$$\mathbf{C}'(\Psi) = \frac{\mathrm{d}\mathbf{C}(\Psi)}{\mathrm{d}\Psi} \quad . \tag{5.18}$$

In the subsequent *update step* the parameters of the Gaussian posterior PDF are computed recursively at each iteration. The posterior mean is computed as follows:

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left(\mathbf{z}_k - h(\hat{\mathbf{x}}_k^-, \mathbf{0})\right) \quad . \tag{5.19}$$

The posterior covariance and Kalman gain can be computed, as shown in Eq. (2.31), by:

$$\mathbf{\Sigma}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{\Sigma}_k^- \quad , \tag{5.20}$$

$$\mathbf{K}_k = \mathbf{\Sigma}_k^- \mathbf{H}_k^T (\mathbf{H}\mathbf{\Sigma}_k^- \mathbf{H}_k^T + \mathbf{\Sigma}_z)^{-1} \quad , \tag{5.21}$$

with the Jacobian matrix of the observation model:

$$\begin{aligned} \mathbf{H}_k &= \left.\frac{\partial h(\mathbf{x}_k^-, \mathbf{0})}{\partial \mathbf{x}_k^-}\right|_{\mathbf{x}_k^- = \hat{\mathbf{x}}_k^-} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \quad . \end{aligned} \tag{5.22}$$

The other covariance matrices, as defined in Eq. (2.29), are:

$$\mathbf{\Sigma}_z = \mathrm{diag}(\begin{bmatrix} \sigma_{w,x}^2 & \sigma_{w,y}^2 \end{bmatrix}) \quad , \tag{5.23}$$

$$\mathbf{\Sigma}_x = \mathrm{diag}(\begin{bmatrix} \sigma_{s,x}^2 & \sigma_{s,y}^2 & \sigma_{s,\Psi}^2 \end{bmatrix}) \quad . \tag{5.24}$$

5.2.3 Discretisation of Absolute Location Information

The following section shall now present an approach to transfer the precise location information from location estimation methods to formats more appropriate for context aware systems.

Although there are services which need precise absolute location, proactive services depending on location cannot depend on absolute location coordinates. If some service is started only given a concrete location coordinate it is likely to be never executed, as coordinates are too fine grained. Instead, regions have to be used which share common semantics: symbolic location.

This fact not only holds for proactive service executing, but also for inference where higher level information is to be inferred from location information, and for learning of preferences or inference rules. If the information about location is too fine grained (inherently to coordinates), no common patterns can be found and learning and inference quality would suffer, respectively.

Hence, a first step to making location information more usable in context aware systems is the translation into symbolic location without losing precision, uncertainty information and without significant computational load. This is shown in section 5.2.3.1. The second step, shown in section 5.2.3.2, is the transformation of meta information from the location estimation techniques to meta information usable for context inference.

5.2.3.1 Adding Semantics to Locations

The core of the transformation of coordinates to symbolic locations is the description of relations between absolute and symbolic information: maps.

Maps

Maps are representations of a portion of the earth's surface, showing how things are related to each other by distance, direction, and size. Digital maps are offered in either vector or raster formats. A special kind of maps are floor plans – simple representations of rooms, doors, etc. from a top view. Walls, doorways, windows and other features of buildings are often drawn to scale and everything appears flat.

All these kinds of maps are used in the literature to improve the performance of position estimation, e.g. in [168, 144, 193]. In particular walls, windows, doors, path ways and corridors are used to match the estimated pedestrian movement and accordingly improve the accuracy and stability of the position estimation process. The representation of the vector floor plan as XML used in [162, 164] is shown in Figure 5.7. Additionally, outdoor features from satellite raster maps (such as buildings locations, path ways, bushes, walkable and non-walkable areas) are used to match the pedestrian movement outdoors [190].

Representation of Maps in R-Trees

In Geographic Information System (GIS) research, methods have been developed to efficiently capture, store, analyse, manage and present data with reference to geographic location data. One outcome is the *R-tree*. This tree data structure is used for efficient access to multi-dimensional spatial data, for instance maps.

An R-tree [158] is a depth-balanced tree data structure designed so that a spatial search requires visiting only a small number of nodes. The R-tree splits the space of areas to be stored, using hierarchically nested, and possibly overlapping rectangles. The "R" in the R-tree stands for "rectangle". If the item to be stored is not a rectangle parallel to the coordinate system, a *minimum bounding rectangles* (MBR) (also called *bounding box*) is stored.

An example R-tree is shown in Figure 5.8. Every node contains an array of x entries with $x \in [m; M], m \in \left[0; \frac{M}{2}\right]$. Each entry of a non-leaf node stores a pointer to a child node and the bounding box of all entries within this child node (represented by its number in the figure). Each entry of a leaf node stores an identifier of the actual data element and its bounding box. The insertion algorithm ensures that nearby elements are inserted in the same node, allowing for the efficient search.

R-trees are commonly used in position estimation frameworks to restrict the search space to areas that are in the vicinity of the tracked user, and to discard *outlier* measurements. This is important to save computation time and complexity. For example, a Wi-Fi access point that is five kilometres away from the tracked user should not affect the position estimation process.

In [191] for instance, RFIDs in the vicinity of a pedestrian are read during each time step of a particle filter and used in the position estimation. Additional RFID measurements received from RFID tags which are not in the vicinity of the pedestrian are ignored.

Maps are suitably represented with an R-tree by the storage of its elements in a spatial order. Rooms, walls, doors can be found easily like that when they are relevant, i.e. in the vicinity of the user. Knowledge about their type can be encoded in the elements' identifier.

Combining Locations with Maps to Symbolic Location

Maps can be used to transform the outcome of location estimation into a symbolic location. Current work integrates them already in the absolute location estimation process

```
<walls>
    <figureAdjustment>
        <!-- 3 points from the floor plan assuming (0,0) is
             the left bottom corner of the floor plan -->

        <coordinateA><x>0.0</x><y>0.0</y><z>0.0</z></coordinateA>

        <coordinateB><x>-52.0</x><y>20.80</y><z>0.0</z></coordinateB>

        <coordinateC><x>0.0</x><y>20.80</y><z>0.0</z></coordinateC>

        <!-- the cartesian real coordinates of the 3 above points-->
        <coordinateAReal>
             <x>0.0</x>
             <y>0.0</y>
             <z>0.0</z>
        </coordinateAReal>

        <coordinateBReal>
             <x>-8.46788</x>
             <y>-57.0808</y>
             <z>0.0</z>
        </coordinateBReal>

        <coordinateCReal>
             <x>19.1737</x>
             <y>-10.46144</y>
             <z>0.0</z>
        </coordinateCReal>
    </figureAdjustment>

    <wall>
        <wallName>wall1</wallName>

        <!-- additional boarder when creating a bounding box for the wall-->
        <additionalBoarderMeters>3</additionalBoarderMeters>

        <coordinateA><x>0.0</x><y>0.0</y><z>0.0</z></coordinateA>
        <coordinateB><x>-52.0</x><y>0.0</y><z>0.0</z></coordinateB>
    </wall>
    <wall>
        [...]
    </wall>

   [...]
</walls>
```

Figure 5.7: Representation of a map with walls preventing crossing used in [168] for location estimation.

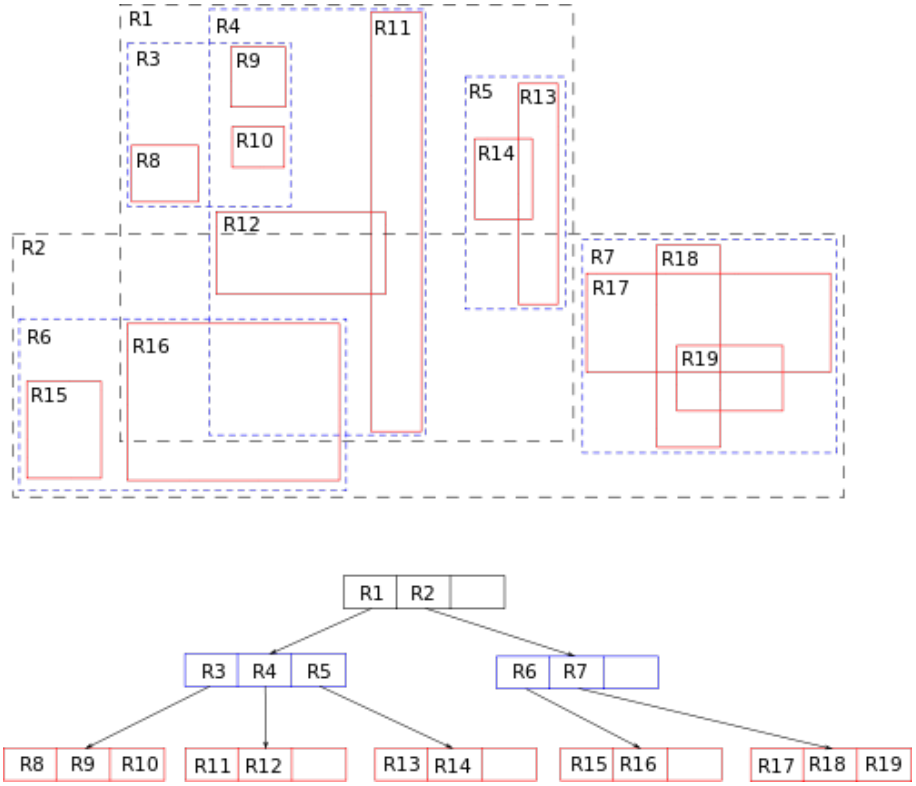


Figure 5.8: Example of an R-tree for 2D rectangles.

to match the pedestrian movement to the map, for instance Krach and Robertson from DLR in [168], Beauregard in [144], and Woodman in [193] apply this successfully with particle filters (see section 2.3.6.3). The advantages are:

- Increased accuracy and stability of the estimator. Particles are prevented from performing motion that is not consistent with the map, e.g. from crossing walls or entering restricted areas.
- Support for noisy or erroneous measurements since it restricts the accessible areas.
- Support for coping with problems of physical sensors like bias and drift.

In more details, the work from DLR includes maps in the following way:
The Sequential Bayesian Positioning Estimator is based on particle filtering and combines several noisy and heterogeneous sensors which provide information about the user's position. Information sources used are GPS, foot mounted INS, electronic compass, RFIDs and altimeter. Enhanced pedestrian motion models are used as prior models to enhance the position estimation process [165]. Currently the position provided is an absolute position using the East, North, Up (ENU) earth based co-ordinate system. In the ENU representation, the east represents the x axis, the north represents the y axis and the up represents the z axis. No knowledge regarding the user's symbolic location is estimated.

The work at DLR [168] uses a mixture between a *Likelihood Particle Filter (LPF)* and a standard particle filter, cf. section 2.3.6.3. Including maps based on R-trees the weighting process assigns low weight to particles which would cross walls since the last step. Therefore all possible borders which are in the vicinity of the tracked user between the last and the proposed position of the particle are retrieved from the R-tree and checked.

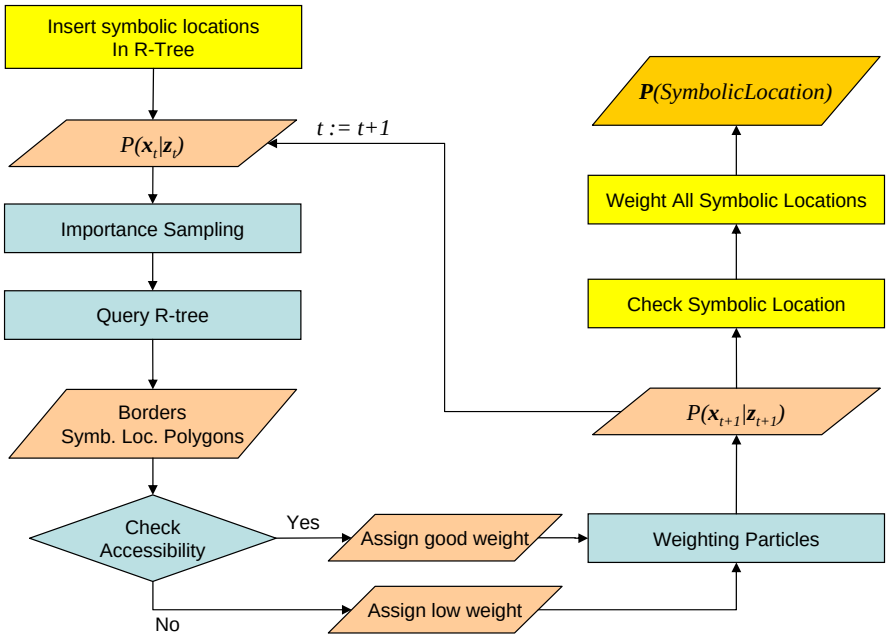


Figure 5.9: Sequential Bayesian estimator of the work in [168] extended to estimate absolute and symbolic location. The steps marked in yellow constitute the extension for symbolic location.

This last step is extended to efficiently compute the symbolic location, as shown in the flow diagram of Figure 5.9. This work proposes to add also symbolic locations like rooms (polygon shaped) to the R-tree (i.e. their bounding boxes). The particle filter is used as before to sample and to weight the particles. However when querying the R-tree in the weighting process for borders, also the symbolic locations in the vicinity of the particle are retrieved. In this step, no extra computational cost arises.

As the R-tree however contains only the bounding boxes of the symbolic locations, also wrong rooms might be returned, every time when a particle is situated within the bounding box, but outside the area of the symbolic location, see Figure 5.10. To resolve this problem, the proposed implementation directly checks with all results of the R-tree query, if the particle's location is really within the symbolic location. This process adds computational costs, but still is comparably efficient, as only those symbolic locations have to be checked whose bounding boxes are around the particle.

When the correct symbolic locations have been identified for every particle, a weighted sum is calculated to find the estimated symbolic location. Their identifiers contain further information about hierarchies and relations among them which can be exploited by the requester. For instance it can contain – in addition to the room name – also the building name, post code, country etc. Given a clearly defined hierarchy of such meta information, it can be avoided that overlapping symbolic locations are returned.

The same process for the identification of the current room with R-tree and individual verification of the results can also be used for all other methods for determining absolute location, e.g. Kalman filters. Instead of combining the usage of the R-tree during location estimation with the lookup of the current room, the R-Tree would be used then after the determination of the most probable (MAP) absolute position and return the room of the most probable absolute location.

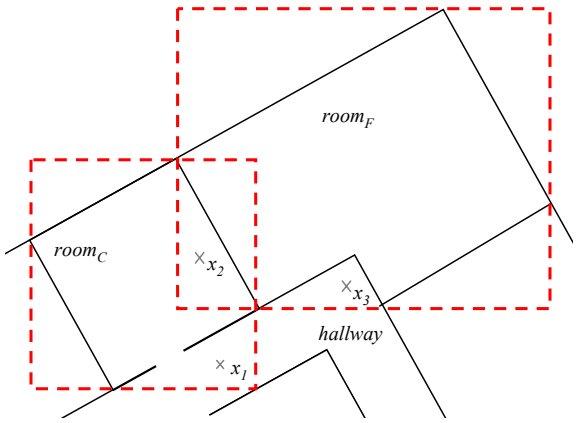


Figure 5.10: Locations x_1 and x_3 are situated within the bounding boxes of a $room_C$ and $room_F$, but not within the rooms themselves. Location x_2 in contrast lies within the bounding boxes of both rooms, but lies in reality only in $room_C$. The returned results of the R-tree query therefore have to be double-checked.

5.2.3.2 Transferring Meta-Knowledge into the Location Context Information

The approach presented above shows how the current symbolic location can be specified, i.e. how from raw data context information can be gained. In order to fully specify context information however, also the meta information has to be set, in the present case hence the meta information has to be transferred from the absolute to the symbolic location.

The most important meta information are the acquisition time and the (un-)certainty which is particularly relevant for Bayesian high level context inference. While the time stamp of the information acquisition can be set easily when calculated in real-time, specifying the uncertainty is more challenging.

Both, the particle filter from the preceding section and the Kalman filter from section 5.2.2, are Bayesian estimators and their estimated posteriors come along with measures for the uncertainty. These measures however differ from each other, as well as from a discrete probability distribution representing the uncertainty in a discrete random variable which shall represent the context attribute *SymbolicLocation* in high level context inference.

In the following the transfer of the uncertainty (meta) information shall be described for particle filters and Kalman filters.

Particle Filter:

Including the transfer from absolute to symbolic location in the particle filter, as shown above, has the advantage that not only the symbolic location of the MAP location is found, but the symbolic location for every particle.

The necessary probability distribution over the found symbolic locations can also be determined easily based on the weights of the particles. Let $\mathbf{X}_t$ denote the position, in particular the latitude and longitude coordinates, estimated by a particle filter using at time t the particles $\mathbf{x}_{i,t}, 1 \leq i \leq N_S$ with the number of particles N_S. If there is a subset of indices $\mathbf{L} \subseteq \{1, ..., N_S\}$ where all particles with indices from this subset are at a symbolic location sl, $\mathbf{x}_{l,t} \rightarrow sl \quad \forall \quad l \in \mathbf{L}$, then the probability of the $SymbolicLocation = sl$ can be calculated as follows:

$$P(SymbolicLocation = sl) = \sum_{l \in \mathbf{L}} w_t^l \ , \tag{5.25}$$

where w_t^i is the weight of particle i at time t according to Eq. (2.34).

Kalman Filter:

The solution for the Kalman Filter proposed in section 5.2.2 is not as straightforward. The outcome of an update step of a Kalman filter at time k is a mean vector $\mathbf{x}_k$ (Eq. 5.19) of the estimated random variables $\mathbf{X}_k$ and a covariance matrix $\mathbf{\Sigma}_k$ (Eq. 5.20).

Hence the information of the posterior covariance matrix has to be transformed into discrete probabilities for the relevant symbolic locations. As a first step therefore, the *relevant* symbolic locations next to the MAP symbolic location have to be identified, with the help of the covariance matrix.

The idea proposed in this work is that the variance of the variables represented in the matrix defines the relevance. In case of 2D positions, these are latitude and longitude, as represented in $\mathbf{R_k}$ above in Eq. (5.7). The variances $\sigma^2_{R_{ik}}$ for the components R_{ik} of $\mathbf{R_k}$ are found in the diagonal of the submatrix of $\mathbf{\Sigma}_k$ related to $\mathbf{R_k}$.

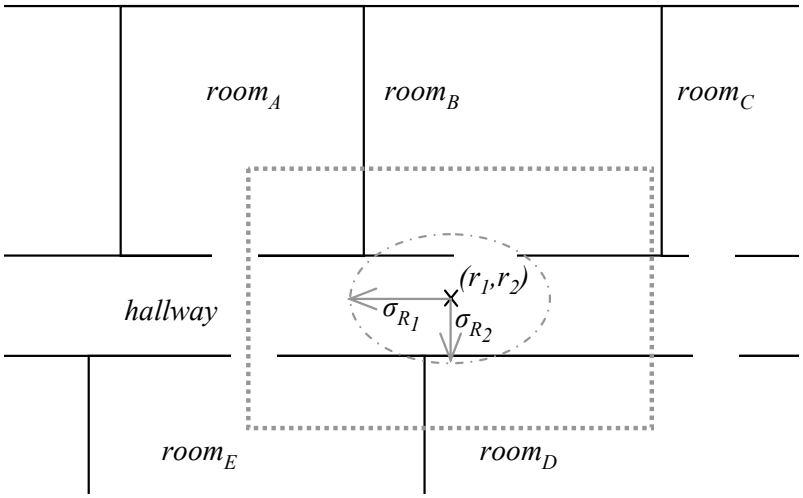


Figure 5.11: A floor plan of an office building, in which a Kalman filter has estimated the user's current position to be (r_1, r_2) with standard deviations $\sigma_{R_1}, \sigma_{R_2}$, lying in the *hallway*. The figure shows, dotted in grey, the area of relevance A for the estimated position determined by $(r_1 - 2\sigma_{R_1}) \leq R_1 \leq (r_1 + 2\sigma_{R_1})$ and $(r_2 - 2\sigma_{R_2}) \leq R_2 \leq (r_2 + 2\sigma_{R_2})$. It contains $room_A, room_B, room_D, room_E, hallway$, but not $room_C$.

Knowing that $P(\mu - 2\sigma \leq X \leq \mu + 2\sigma) \approx 0.954$ for normal distributed X with mean μ and variance σ^2, the multi dimensional 2σ environment of the position related vector $\mathbf{R_k}$ is the relevance space. In the usual case of two dimensional positions shown in Figure 5.11, this environment represents the area A with $P(\mathbf{R_k} \in A) > 0.9$. The roughly 10 % remaining are neglected in order to restrict the relevance and hence to reduce the computational complexity.

This area of relevance A is queried then in the R-tree and the relevant symbolic locations are found. As the second step, the probability for each symbolic location has to be calculated. Every symbolic location sl_i has borders determining its area and the probability $P(SymbolicLocation = sl_i)$ is determined then by the cumulative probability density $F(\mathbf{R_k})$ in the area of sl_i.

Although in the multivariate case, there is no closed form for $F(\mathbf{X})$ [66], there are a number of algorithms that estimate it numerically which can be used, c.f. the work of Liu in [172].

The result finally has to be normalised so that $\sum_{sl_i} P(SymbolicLocation = sl_i) = 1$.

5.2.4 Modelling of Symbolic Location in Bayesian Networks

The last section has shown how to translate absolute location into symbolic location given a map. Modelling symbolic locations in Bayesian networks however is challenging for the following reasons.

- Symbolic locations are not mutually exclusive.
- The number of possible symbolic locations is unlimited.

Careful modelling of the information available in the R-tree in agreement with the requirements of the high level context inference rule can solve this problem. Such solutions however are not appropriate for generic applicability and usage of the location and context inference necessary in ubiquitous computing.

5.2.4.1 Generic, Modular Modelling of Symbolic Location Projections

A straightforward solution to make the applicability more general is to model all dependencies in the Bayesian inference rule. This model is based on absolute location and its sensors which are causally influenced by the absolute location.

Symbolic locations are projections of the continuous, multi dimensional absolute locations onto a discrete set of nominals, representing the different symbolic locations. As the target models are Bayesian networks, the different nominals also have to be mutually exclusive and their set has to be finite, in order to be able to represent it in an R-tree.

With these assumptions, symbolic location is a surjective function of the current location vector $\mathbf{x}$ and the map m:

$$sl = SL(\mathbf{x}, m) \tag{5.26}$$

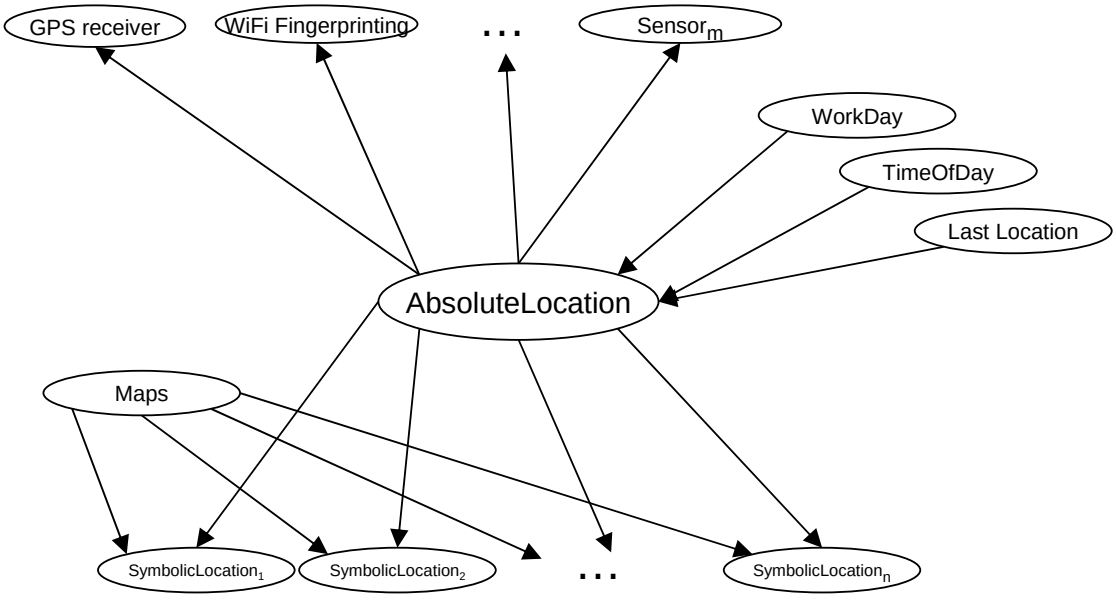


Figure 5.12: Generic model of absolute locations, symbolic locations and location sensors in a static Bayesian network. It consists of the continuous variable *AbsoluteLocation*, different sensors $Sensor_i$ influenced by the symbolic location and a set of projections to the discrete RVs $SymbolicLocation_i$, as well as other external information which adds knowledge about the current absolute location, such as $TimeOfDay$.

The representation of non-mutual exclusive, hence overlapping symbolic locations is possible with a set of functions SL_i each referring to different mappings. In terms

of Bayesian random variables, these functions can be modelled like in section 5.2.2.4: A symbolic location is represented by a random variable with the parents *Map* and *AbsoluteLocation*, as shown in Figure 5.12. Different functions for overlapping symbolic location are consequently modelled as different random variables.

The problem with this model is its computational complexity. *AbsoluteLocation* and at least some of the sensors represent continuous variables. Moreover the number of projections to symbolic location which can possibly be relevant for some high level context is if not unlimited then at least very high.

To come up with a tractable solution for resource constrained devices, this work proposes a flexible and modular approach. As a first step, the dynamic inference of the continuous RV *AbsoluteLocation* is to be decoupled from the further usage of symbolic location, as shown in an example in section 5.2.2. The second step is the construction of pluggable inference modules for each random variable $SymbolicLocation_i$ which can be used by the designer of the high level context inference rule.

In such pluggable inference modules, the estimated absolute location is used to find, with the help of an R-Tree query, the correct symbolic location – applying the process described in section 5.2.3.

To maintain the dependency structure of figure 5.12, all RVs concerning symbolic location used in parallel in an inference network have to be set to soft evidence, which has the same d-separation properties like *AbsoluteLocation* with hard evidence in at least one $Sensor_i$ RV.

To represent the common dependency of all symbolic location RVs to *AbsoluteLocation*, the respective soft evidence always has to be updated in parallel. The corresponding inference engine can ensure this with the help of the context model (see section 2.1.2 on page 18). Evidence in a random variable always represents a context information, the random variable's name specifies thereby the context type, for example $WorkPlaces$. All random variables concerning the symbolic location however have the same *aspect* from the ASCi-Model, namely `SymbolicLocation`.

Hence when a context inference rule is to be evaluated which contains a RV of aspect *SymbolicLocation* the following steps are executed:

1. The respective inference modules, specifying the correct context type are retrieved from a repository.

2. The R-tree part of a retrieved inference module is used in the dynamic inference module for absolute location, if a particle filter is used.

3. All modules are evaluated incorporating the current posterior of the dynamic location estimation and produce a probability distribution for its output context type.

4. The outcomes of all modules are introduced as soft evidence into the respective random variables of the context inference rule.

The inference modules for symbolic location thereby can be provided by a central server in tiles concerning different regions, like today's map tiles for online satellite map applications. In addition also personalised symbolic location modules are of importance, for instance to specify the symbolic locations for daily routines $atWork$, $atHome$, $atSports$, $inTheCity$ which depend on the user himself. Building up personalised modules for a user would not be difficult using a graphical map for location tagging. An R-tree can be populated then from the tagged areas.

5.2.4.2 Tree Structured Symbolic Location Projections

A special case of overlapping projections from absolute to symbolic location is where each symbolic location is overlapped by a number of sub-locations which do not overlap with other symbolic locations. These projections can be represented by a tree like the one shown in 5.13.

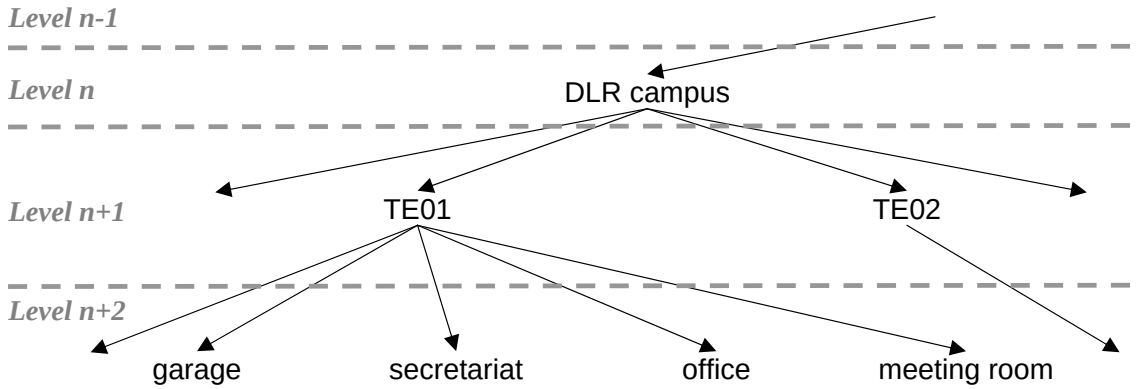


Figure 5.13: Part of a location hierarchy with the levels of districts (Level n), buildings (Level n+1), and rooms (Level n+2).

This tree has the following property for a symbolic location sl with n sub-locations sl_i:

$$P(SymbolicLocation = sl) = \sum_{i=1}^{n} P(SymbolicLocation = sl_i) \tag{5.27}$$

In other terms: the whole area of sl is completely partitioned in sublocations sl_i.

Modelling such projections in random variables would show the less fine grained, lower level projections independent from the absolute location, as their behaviour can be conclusively determined by the more detailed projection. Such modelling however would be too cost intensive, because the less fine grained RV does not add any new knowledge to the network. This redundancy could be avoided by using always only the projection of the highest level.

Higher level projections however contain many different sub-locations which often will not be necessary. In terms of Figure 5.13, when a user is in building *TE01*, all rooms of *TE02* are irrelevant. Having modelled them in the BN means, however, that any parent and child node causes the addition of conditional probabilities for the sub-location either in the node itself or the child node.

The property from Eq. (5.27) allows a more intelligent solution. The value range of the *SymbolicLocation* RV can be dynamically formed based on the current relevant sub-locations. The evaluation of the R-tree based on the estimated absolute location gives the highest level of detail necessary. The value range of *SymbolicLocation* then has to contain all relevant sub-locations, but can collapse those sub-trees which have no relevance.

Based on the fact that the different levels represent larger areas between which human users do not change too frequently, the CPTs of the nodes in the static BN can be reduced considerably and do not have to be modified too frequently. Section 5.4.3 will be more detailed with regards to the mathematical operations necessary for the adaptation of the CPTs.

5.2.5 Proximity Determination

A crucial factor for context-awareness which is based on location is *proximity*. The services and people which are close to a user influence his options and actions. The determination of proximity is challenging for the following reasons:

1. The determination of "near" and "far" is relative and changes depending on context.
2. A user has proximity relations to all other existing entities, i.e. services, actuators, objects and persons.
3. The values of these relations constantly change.
4. The distance to other entities is usually not linear, as the *line of sight* is often blocked.

The present research offers an approach to provide proximity information for a user to a context aware system. The approach is based on the same concepts as the last sections, precise absolute positioning, symbolic location using maps and R-trees and modular on-demand inference.

What is "near" and "far"?

As mentioned above, these terms depend on the context of their application, in particular on the service requiring this information. The service requesting proximity information can specify a range of proximity, interpreted as a distance in metres to the user's location. Easier is the determination of "nearest", which can be always solved without additional information of services.

Like this, determination of proximity boils down to a calculation of distance and comparing to a fixed threshold.

There are proximity relations to all entities.

Calculating the distance to all entities would exceed the computational capabilities of resource constrained devices in a reasonable response time. Consequently the entities have to be further specified and limited. Again, the service requesting the information can provide the most useful information, by specifying for instance *doors*, *windows*, *printers* or *friends* of which the proximity is relevant.

To further limit the search space, the number has to be reduced more, whereby static (like *doors*, *windows*, *printers*) and mobile entities have to be distinguished.

The R-tree of the current location helps to limit the search space for static objects, assuming that all static objects are inserted as map elements. The defined proximity distance allows to query then the R-tree for a circular area around the user's location with the proximity distance as radius.

This approach is not viable for dynamic objects, as the permanent insertion and deletion of objects in the R-tree would destroy its performance. Dynamic objects however are in particular persons, from which we can assume that they track their own position to allow for instance proactive context aware services. In this case, they also have a symbolic location. If persons are close enough to be relevant for proximity estimation, then a more or less high level of symbolic location (compare Figure 5.13) will have the same values.

Proximity is constantly changing.

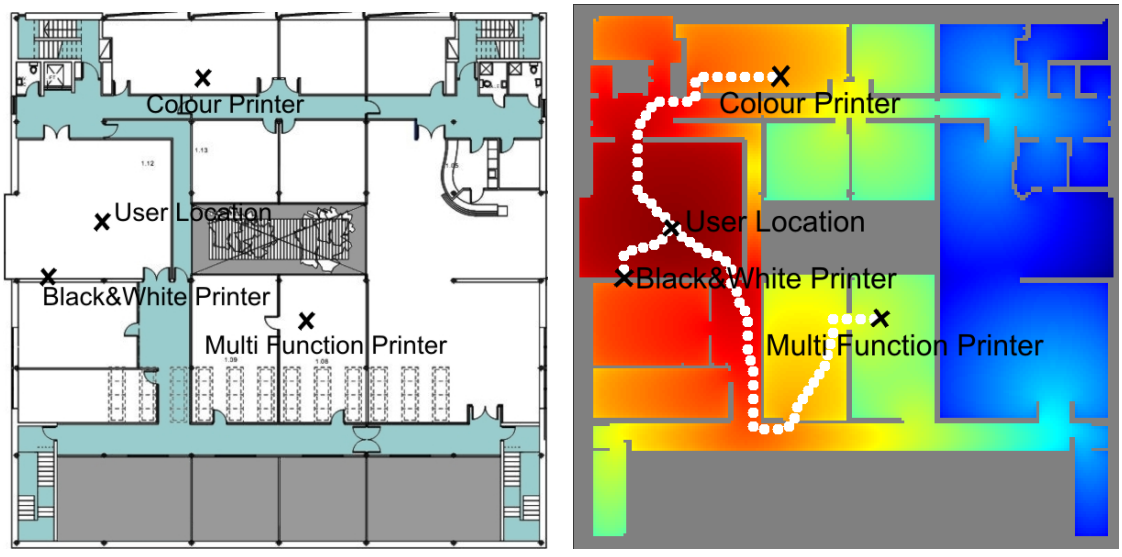
Also the permanent calculation of changing distances would overburden mobile devices. The subscription (compare section 5.1.2.1) hence has to be prohibited and can be queried only then on demand, when other necessary conditions are already met. This logic again has to be implemented by the service provider.

Proximity depends on the environment settings.

The problem of determining a proximity has now been reduced to distance calculations between the user's location and the locations of a limited set of relevant entities. Simply applying a straightforward Euclidean metric in 3D space will usually fail, since non-linearities, such as walls, in the true practical proximity are not considered. A true proximity metric should be proportional to something like the time taken to reach the destination or the distance travelled.

In [140] a mobility model for pedestrians was introduced that applies a gas diffusion algorithm to estimate a path between two points in a given floor plan layout. The algorithm works by virtually emitting gas from the destination that propagates through the open portions of a building layout and becomes absorbed by walls. Any point in the building experiences a certain concentration of the gas that is proportional to the distance to the emitter. Computation of the gas source is straightforward [163] and the results may be cached in a database.

This work proposes to invert the gas diffusion approach to measure a number of distances from a given point. The user's location is used as position of the gas emitter. To calculate the distance, the shortest path to the destination is identified by following the steepest gradient of gas density to the destination, known from the calculated density matrix. In a map with well defined coordinate system, the calculation of the length of the path is straightforward.



(a) (b)

Figure 5.14: The floor plan of the Arclabs building of TSSG, Waterford Institute of Technology, with marked locations of a user and three different printers (left). On the right side, the proximity of the three printers to the user is measured applying a gas diffusion model.

The result is shown in Figure 5.14. A service prints documents always on the printer closest to the user. In an office building like the one shown in Figure 5.14 (a), three usable

printers have been identified, one in the board room, one in the IT administration office, and one in the printing room. As the user is currently situated also in the board room, the printer there is recognised as the nearest one (the colour of the gas density is represented darkest in Figure 5.14 (b)) and used. If black and white was not sufficient, the printers in the IT administration office and the printer room would be the options. Although the Euclidean distance to both printers is almost identical, the path to the printer in the printer room is considerably longer. Hence the one in the IT administration office would be used.

5.2.6 Discussion

A lot of research has been undertaken on location based services, on efficient access to spatial information and on maintenance of location information. Often however usage of location information in context aware environments has been neglected.

This work therefore has provided the complete chain from the fusion of different location information to its conversion into symbolic location and the incorporation in high level context inference, as well as the derivation of proximity from it.

The proposed method for absolute location estimation is using a Kalman filter. This is the most efficient way in terms of computation and largely independent from infrastructure data, which makes this approach particularly relevant for resource constrained mobile devices.

Also the second widespread location estimation method, a particle filter, is used. Its advantages include the more general applicability and the inclusion of maps which are necessary for the determination of symbolic location. This leads to very good accuracy results and a very efficient determination of symbolic location.

As already postulated in the requirements of section 3.3, a modular approach for the composition of high level context inference rules ensures efficiency for the incorporation of symbolic location information. Only the relevant information for an inference goal is computed, while the independence relations of a Bayesian network encompassing all location information is maintained.

The approach for calculating proximity information of a user relies on the proposed methods for absolute and symbolic location. Due to some restrictions (no continuous inference) and limitations of scope (by symbolic location and explicit requirements to service providers) a tractable solution using a gas diffusion model and, again, maps can be proposed. This method is not too resource efficient, but provides – in contrast to simpler methods – a realistic estimation of distance which is fundamental to detecting proximity. The efficiency shortcomings (for every new location a matrix for the gas diffusion, proportional to the size of the area, has to be computed) are attenuated however by the proposed restricted usage of this method.

The collective of the proposed methods for location information allow for the generic and resource efficient applicability in ubiquitous computing, and for high level context inference in particular.

5.3 Recognition of Human Motion Related Activities

The last section has shown how to infer precise location with Bayesian approaches from multiple sensors and how to integrate it in context inference as symbolic location. In some situations when there is a single reliable sensor for absolute position, there is no need for inference and even transformation to symbolic location might not be necessary when some information source provides already symbolic location.

There are however also context aspects that cannot be measured directly by a sensor and which always have to be inferred. One of them is the human physical activity: the information, whether a person is *sitting*, *standing*, *walking*, *running*, *jumping*, *falling* or *lying*.

This section shows how to infer this context information from acceleration and turn rate measurements in real-time on resource constrained devices. From a big amount of high frequent raw data, the presented approach produces less frequent, compact context information that can be used by different context consumers. This specifies hence besides location a second context inference module usable for high level context inference.

The remainder of this section first discusses the relevance of motion related context information, then describes the available raw data and the features that are relevant for the investigated activities, before different designs of the context inference rules, i.e. the Bayesian classifier, are presented according to the publications of the author et al. in [211, 237, 209, 210].

5.3.1 Importance

Next to location, one of the most important context attributes defining the user's situation is his physical activity. Its importance is not only visible in the large amount of research performed in this area (cf. section 4.3), but also in the scenarios in the introduction from page 3 onwards.

The variety of use cases where human motion related activity is necessary reflects the fact that 15 of the 35 context aware services in section 1.2 depend on it. While such situation is useful in situations at work (section 1.2.1) or in entertainment (section 1.2.2), it is absolutely vital in other environments such as:

- Ambient Assisted Living:
  Situations like the one described in section 1.2.3 are getting more and more frequent. Thereby the elderlies' health has number one priority. Prerequisite for noticing deviation from normal life or accidents is continuous monitoring of the activity. Only like that, falls or also degrading mobility can be detected and severe consequences avoided.

- Safety Critical Missions:
  Some jobs have immanent risks caused by the environment in which they are executed, for example firemen, police in exertion or disaster relief experts. When such helpers' safety is endangered, their general demand for privacy is overridden and it is useful to have a supervisor informed about the current status. Only with current information about a helper's situation and physical status in particular, measures to ensure his safety can be taken.

- Pervasive Health:
  Topics of pervasive health are more and more relevant. Governments and health insurers have as high an interest in people's health as people themselves. Gymnasiums, but also video game consoles and social networks come up with applications

for monitoring one's health and in particular the degree of physical activity and performance in sports. Based on the measured data, energy consumption can be calculated or hints for further training are given in order to improve the personal health.

In addition to its direct usage by services like those described above, it is also of high importance for describing the user's situation. Together with location, the physical activity has big influence on high level context, like the activities of daily life as described in section 3.2.1.

5.3.2 Human Motion and Measured Inertia

This section explains the raw data available for recognition of human motion related activities, their preprocessing and their quantisation. After these steps, every state of the preprocessed values, the so called *features*, adds value to the classification.

5.3.2.1 Hip Mounted Inertial Measurement Unit

This work uses a hip mounted inertial measurement unit (IMU) to infer motion related activities. With the IMU, different sensor types (accelerometers and gyrometers), i.e. multimodal sensors, are used at one place of the body, instead of distributing several sensors of the same type across the body. According to Choudhury et al. in [205] "studies have shown that the information gained from multimodal sensors can offset the information lost when sensor readings are collected from a single location".

The hip has proven to be the most suitable placement, at it is close to the centre of gravity of the human body and is sensitive to any movement of the trunk and both legs. Using only one sensor moreover is considerably less obtrusive, one of the main factors for the success of ubiquitous computing. Placing the sensor at the hip, a MEMS based IMU could also be integrated in the belt buckle.

The IMU used in this work, an xsens MTx as displayed in Figure 5.15 was set to provide measurements with 100 Hz. It contains three orthogonal accelerometers, gyrometers for yaw, pitch and roll and three orthogonal magnetometers. The latter however do not add value to activity recognition and shall be neglected in the remainder of this section. Further on, the IMU has an embedded temperature sensor for internal compensation and an embedded processor capable of calculating the orientation of the sensor in real-time which provides calibrated 3D linear acceleration, turn rate and magnetic field data [241].

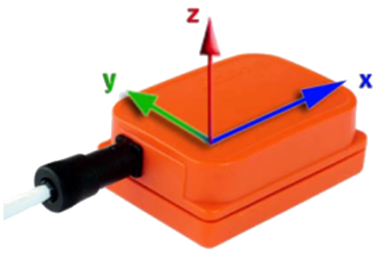


Figure 5.15: xsens MTx sensor with frame representation from [241].

The orientation is calculated by the fusion of the sensor output signals using a Kalman filter [231]. Figure 5.16 summarises the information used. The orientation is provided between the sensor frame (shown in Figure 5.15) and the *global frame* (GF), an earth-fixed reference frame defined in Cartesian coordinates.

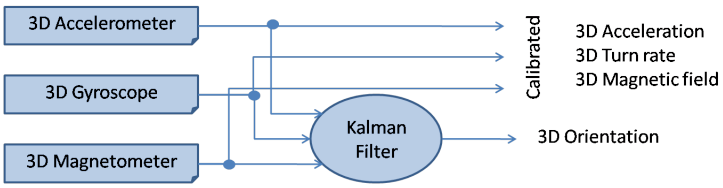
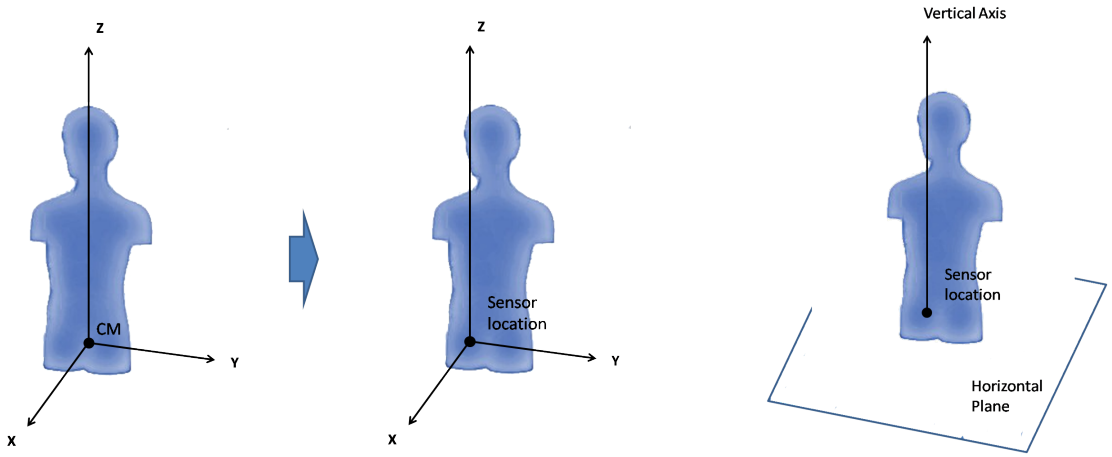


Figure 5.16: The xsens MTx sensor fusion scheme [231]. The fusion of the information of the 3D accelerometers, gyroscopes and magnetometers yields the 3D orientation of the sensor.



(a) The ideal body frame has its origin in the centre of mass of the human body while it is standing upright and does not vary over time. In reality, the origin of the body frame is defined by the sensor location which depends on the individual user's placing.

(b) Final approximation to the body frame: The horizontal plane allows access to all necessary information.

Figure 5.17: Definition of the body frame at the sensor location.

Acceleration and angular velocity are most relevant in relation to the human body, hover, which is why they are (approximatively) converted to the *body frame* (BF). In contrast to the global frame, the body frame follows the movements of the user. The three axes of the body frame are defined to intersect at the sensor location (see Figure 5.17 (a)), the z axis is directed towards the head, while the other axes (x and y) form the plane orthogonal to this vertical axis.

In order to obtain the rotation between the sensor frame and the vertical axis in the body frame, the 3D orientation computed internally in the sensor module can be used (see Figure 5.17 (b)). The exact heading of the human body cannot be estimated unless magnetometers are used, in this work however the information used for inference does not require the direction of the sensor frame. Determination of the horizontal plane allows for the computation of all necessary information.

5.3.2.2 Sources of Information

Related work in biomechanics (e.g. [240]) and accelerometer based recognition of human motion related activities (described in section 4.3) usually identifies the *norm* of the acceleration as the main source of information. This is however only a part of the information characterising human motions. IMUs provide acceleration and angular velocity regarding different reference frames, as well as angular information describing the axes of every frame.

Our approach clusters the 100 samples received per second into windows of different length, ranging from 32 to 512 samples, i.e. from roughly a third of a second to five seconds. This allows for the identification of very short movements (like a fall) up to physical activities consisting of a repeated pattern (like running).

In these windows, the following sources of information are computed from the available raw data. They serve as a basis for the identification of features in this work.

- $|a|$, the norm of the acceleration a, and $|\omega|$, the norm of the angular velocity ω, in the global frame defined as

$$\begin{aligned} |a| &= \sqrt{a_x^2 + a_y^2 + a_z^2} \\ |\omega| &= \sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2} \ , \end{aligned} \tag{5.28}$$

  where a_i and ω_i, $i \in \{x, y, z\}$ is the acceleration and angular velocity respectively at the i-axis.

- $\left|a_h^{BF}\right|$, horizontal acceleration in the body frame, and $\left|\omega_h^{BF}\right|$, angular velocity in the horizontal plane of the body frame,

$$\begin{aligned} \left|a_h^{BF}\right| &= \sqrt{(a_x^{BF})^2 + (a_y^{BF})^2} \\ \left|\omega_h^{BF}\right| &= \sqrt{(\omega_x^{BF})^2 + (\omega_y^{BF})^2} \ , \end{aligned} \tag{5.29}$$

  where a_i^{BF} and ω_i^{BF}, $i \in \{x, y\}$ is the acceleration or angular velocity, respectively, at the i-axis of the body frame.

- a_v^{BF}, vertical acceleration in the body frame, and ω_v^{BF}, angular velocity of the vertical axis in the body frame,

$$\begin{aligned} a_v^{BF} &= a_z^{BF} \\ \omega_v^{BF} &= \omega_z^{BF} \ , \end{aligned} \tag{5.30}$$

  where a_z^{BF} and ω_z^{BF} are the acceleration and angular velocity along the z-axis of the body frame.

- a_v^{GF}, vertical acceleration in the global frame,

$$a_v^{GF} = a_z^{GF} \ , \tag{5.31}$$

  where a_z^{GF} is the acceleration measured along the z-axis of the global frame.

Sources of information exploited for activity recognition are furthermore:

- in the temporal domain:
  - Minimum, *min*, and maximum, *max*, values in a window and the range between both.
  - Mean $\bar{x}$ and standard deviation $\sigma(x)$ of a quantity x.
  - Median and mean absolute deviation.
  - The *inter quartile range* (IQR): the range between the 25-percentile and the 75-percentile, hence the range of the 25 % environment around the median.
  - Root Mean Square (RMS): the square root of the arithmetic mean of the squares of information in a window.

  - Integrated value using trapezoidal approximation.
  - Mean crossings.
  - Pearson correlation coefficient ρ between two signals.

- Frequency domain features:
  - Main frequency component (MFC).
  - Spectral entropy and relative spectral entropy.
  - Energy $\hat{E}$ of the signal in a frequency band determined either by a low pass filter (LPF) or a band pass filter (BPF).

5.3.2.3 Features of Human Motion Related Activity

In the terminology of classification, the low level information which helps to infer the higher level context *physical activity* represents a set of *features* of the searched class. In the present case, mathematically, features are functions, mapping the state space of the measurements to a value in the real numbers.

$$f(\vec{E}) \in \mathbb{R} , \tag{5.32}$$

$$\vec{E} = (a_x, a_y, a_z, \omega_x, \omega_y, \omega_z, \epsilon_1, \epsilon_2) , \tag{5.33}$$

where $\vec{E}$ is the vector of measureed evidence, a_i and ω_i are the acceleration and turn rate in direction $i \in \{x, y, z\}$, respectively. $\epsilon_j, j \in \{1, 2\}$ are the Euler angles for the conversion between sensor frame and global frame which could also be replaced by quaternions or a rotation matrix. The basic information obtainable from the IMU's raw data presented in section 5.3.2.2 are used to compute the features.

The objective is to identify features that allow for reliable inference of the motion related activities, independent from the person and the dataset used to train the classifier (i.e. to learn the Bayesian network). The set of features is meant to describe the complete picture of human motions with a minimum number of efficiently computable features, in order to allow efficient computation.

Therefore the criteria for selecting a feature have been the following:

- Justification of the influence with a bio-mechanical and physical explanation
- Discriminative power between the activities in question.
- Efficient computation.
- No redundancy with other features.

Based on the bio-mechanics of human motion and the available sources of information from the IMU, 19 features have been selected that fulfil the criteria described above. The following paragraphs will describe the selected features and show examples, how they discriminate and cluster activities together with other selected features.

Horizontal Acceleration in the Body Frame

Features 1 – 3 use the horizontal acceleration in the body frame, see Eq. (5.29).

$$f_1(\vec{E}) = \max_{128}\left(\left|a_h^{BF}[k]\right|\right), 1 \leq k \leq 128 \tag{5.34}$$

$$f_2(\vec{E}) = \overline{\left|a_h^{BF}\right|}_{128} = \frac{1}{128}\sum_{k=1}^{128}\left|a_h^{BF}[k]\right| \tag{5.35}$$

$$f_3(\vec{E}) = \sigma_{128}\left(\left|a_h^{BF}\right|\right) = \sqrt{\frac{1}{128}\sum_{k=1}^{128}\left(\left|a_h^{BF}[k]\right| - \overline{\left|a_h^{BF}\right|}_{128}\right)^2} \tag{5.36}$$

The maximum horizontal acceleration $\max_{128}\left(\left|a_h^{BF}\right|\right)$ helps distinguishing between static and dynamic activities (see Figure 5.18) and has particularly high values for *falling* (when the body hits the floor). The mean value, shown in Figure 5.22, helps to distinguish the static activities (almost no horizontal acceleration for *standing*, the full gravity for *lying*, and *sitting* in between them) and the standard deviation helps to distinguish the activities *jumping*, *falling* and *running*, in particular in combination with feature 6.

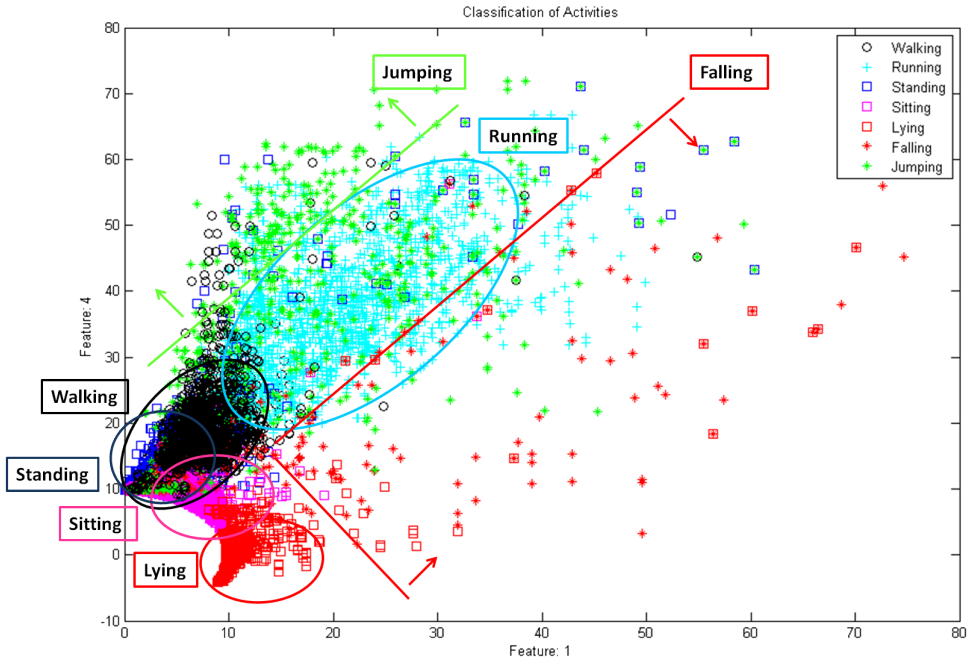


Figure 5.18: Plot for $\max_{128}\left(\left|a_h^{BF}\right|\right)$, feature 1, and $\max_{128}\left(a_v^{BF}\right)$, feature 4. The combination of both allows to distinguish *standing, sitting* and *lying* as the body attitude information is contained in the body frame. The distinction of *falling* and *jumping* is also possible as both activities reach their maximum value in different signals once the body has hit the floor. Also *walking*, *running* and *jumping* can be distinguished.

Vertical Acceleration in the Body Frame

The vertical acceleration in the body frame, the signal defined in Eq. (5.30), is used for the features 4 – 7.

$$f_4(\vec{E}) = \max_{128}\left(a_v^{BF}[k]\right), 1 \leq k \leq 128 \tag{5.37}$$

$$f_5(\vec{E}) = \overline{a_v^{BF}}_{128} = \frac{1}{128}\sum_{k=1}^{128} a_v^{BF}[k] \tag{5.38}$$

$$f_6(\vec{E}) = \sigma_{128}\left(a_v^{BF}\right) = \sqrt{\frac{1}{128}\sum_{k=1}^{128}\left(a_v^{BF}[k] - \overline{a_v^{BF}}_{128}\right)^2} \tag{5.39}$$

$$f_7(\vec{E}) = RMS_{128}\left(a_v^{BF}\right) = \sqrt{\frac{\sum_{k=1}^{128} a_v^{BF}[k]}{128}} \tag{5.40}$$

The maximum value $\max_{128}\left(a_v^{BF}\right)$ helps to distinguish between *jumping*, *falling* and *walking*. The mean value distinguishes *standing*, *sitting* and *lying*, while the standard deviation helps discriminating between all dynamic activities. The root mean square $RMS_{128}\left(a_v^{BF}\right)$ is a good discriminator for the static activities. It can be seen in Figure 5.19 where it identifies reliably *lying*, other static activities and dynamic activities.

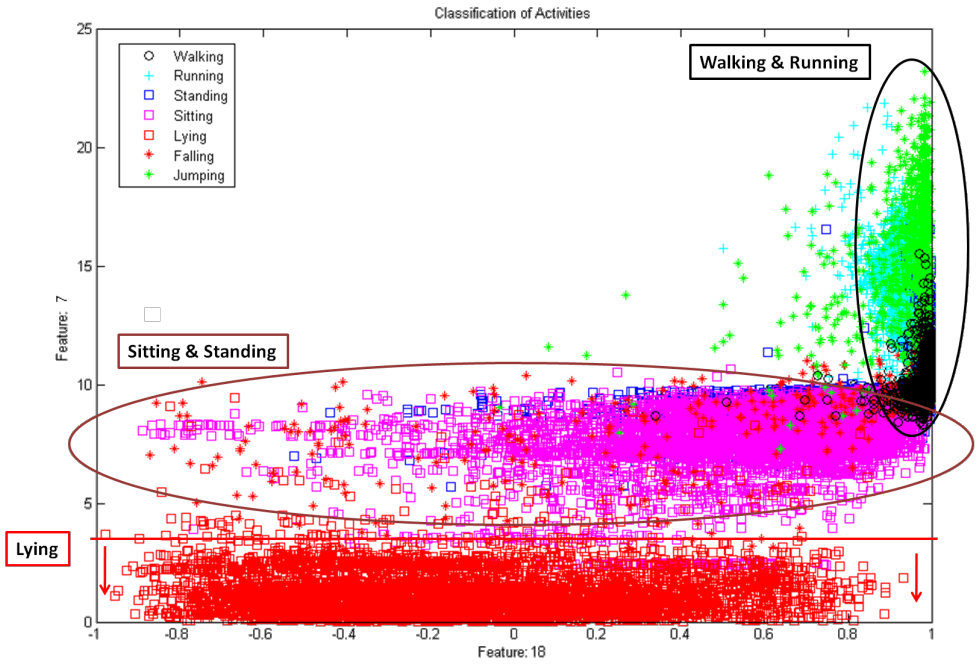


Figure 5.19: Plot of feature 18, $\rho_{128}\left(a_v^{BF}, |a|\right)$ and feature 7, $RMS_{128}\left(a_v^{BF}\right)$. Feature 18 achieves high values if the current motion acceleration in 3D is mainly contained in the vertical axis. Feature 7 is useful for the distinction of *lying* from other activities.

Horizontal Angular Velocity in the Body Frame

Feature 8 is the only feature using angular velocity.

$$f_8(\vec{E}) = IQR_{128}\left(\left|\omega_h^{BF}\right|\right) \tag{5.41}$$

While for other motions, accelerometers can offer sufficiently good discriminators, the recognition of *falling* which is shorter than one second can be improved. A falling body often suffers from a fast rotation which can be measured by the gyroscopes. It is used in the interquartile ranges $IQR_{128}\left(\left|\omega_h^{BF}\right|\right)$ which has a similar meaning as a σ-environment, but relative to the median. As expected, particularly high values are reached for *falling*, which can be seen in Figure 5.20.

Vertical Acceleration in the Global Frame

Also from the vertical acceleration in the global frame, a single feature has been chosen:

$$f_9(\vec{E}) = \overline{a_v^{GF}}_{32} = \frac{1}{32}\sum_{k=1}^{32} a_v^{GF}[k] \tag{5.42}$$

Feature 9 uses the vertical acceleration in the global frame, signal (5.31). Only in the global frame, free fall phases can be easily observed. As these are very short for standard human motions, the shortest window with 32 samples has been chosen. Free fall phases are seldom for falling, but part of every jump. Hence it is a good discriminator for *jumping*, as can be seen in Figure 5.20 where *jumping* and *falling* are reliably recognised.

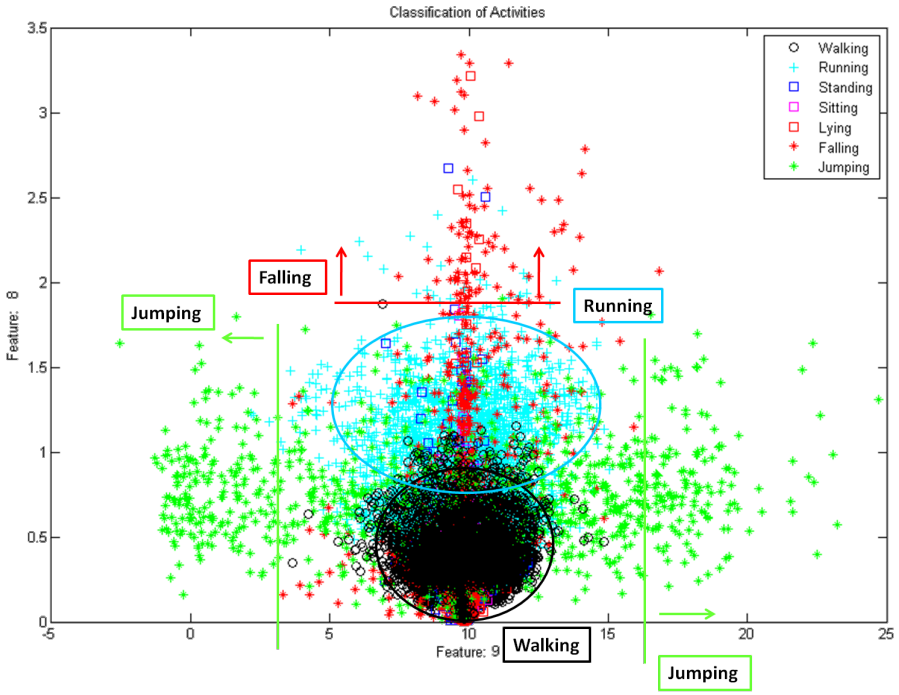


Figure 5.20: Plot of feature 9, $\overline{a_v^{GF}}_{32}$ and feature 8, $IQR_{128}\left(\left|\omega_h^{BF}\right|\right)$. Feature 8 is specially useful for *falling* that implies a rotation of the body in its horizontal plane, feature 9 for *jumping*, as it detects short phases of free fall.

Norm of the Acceleration

Features 10 – 17 are computed from the norm of the acceleration, the signal shown in Eq. (5.28).

$$f_{10}(\vec{E}) = \overline{|a|}_{32} = \frac{1}{32}\sum_{k=1}^{32} |a[k]| \tag{5.43}$$

$$f_{11}(\vec{E}) = \overline{|a|}_{512} = \frac{1}{512}\sum_{k=1}^{512} |a[k]| \tag{5.44}$$

$$f_{12}(\vec{E}) = \sigma_{256}\left(|a|\right) = \sqrt{\frac{1}{256}\sum_{k=1}^{256}\left(|a[k]| - \overline{|a[k]|}_{256}\right)^2} \tag{5.45}$$

$$f_{13}(\vec{E}) = IQR_{128}\left(|a|\right) \tag{5.46}$$

$$f_{14}(\vec{E}) = MFC_{128}(|a|) \tag{5.47}$$

$$f_{15}(\vec{E}) = \hat{E}_{128}\left(\overset{2.85\ \text{Hz}}{\text{LPF}}(|a|)\right) \tag{5.48}$$

$$f_{16}(\vec{E}) = \hat{E}_{64}\left(\underset{1.6\ \text{Hz}}{\overset{4.5\ \text{Hz}}{\text{BPF}}}(|a|)\right) \tag{5.49}$$

$$f_{17}(\vec{E}) = \hat{E}_{512}\left(\underset{1.6\ \text{Hz}}{\overset{4.5\ \text{Hz}}{\text{BPF}}}(|a|)\right) \tag{5.50}$$

Mean values over different window lengths are relevant for short-term and longer, repetitive activities. The standard deviation $\sigma_{|a|}$ helps distinguishing between static and dynamic activities, while the interquartile range $IQR_{|a|}$ is relevant for the distinction between *jumping* and *falling*. The interquartile range is the difference between the 25th and the 75th percentile (where the 50th percentile is the median).

The main frequency component $MFC_{128}(|a|)$, computed by a Fast Fourier Transform, can identify *walking* and *running* and is in particular used to distinguish *falling* and *jumping* from *running*. It is shown in combination with feature 15 in Figure 5.21.

Features 15 – 17 represent the energy of the norm of the acceleration in some particular frequency bands computed by Finite Impulse Response filters, see [237] for details. While the low pass filter below 2.85 Hz in feature 15 is helpful in distinguishing between *walking* and *jumping* or *running*, the band pass filters between 1.6 Hz and 4.5 Hz in features 16 and 17 can help to distinguish between *running* and *jumping* for short and long time activities respectively.

Correlation between Accelerations

The correlation coefficient $\rho_{128}\left(a_v^{BF}, |a|\right)$ (i.e. between the accelerations defined in Equations (5.28) and (5.30)) is used as feature 18.

$$f_{18}(\vec{E}) = \rho_{128}\left(a_v^{BF}, |a|\right) = \frac{\overline{a_v^{BF} \cdot |a|}_{128} - \overline{a_v^{BF}}_{128}\overline{|a|}_{128}}{\sigma_{128}\left(a_v^{BF}\right)\sigma_{128}\left(|a|\right)} \tag{5.51}$$

It can be observed that the norm of the acceleration is mainly determined by the vertical acceleration in the body frame while *walking* and *running* and also *jumping*. This information is used to build a classifier for these dynamic activities with feature 18. *Walking*, *running* and *jumping* have very high values for this feature, while other activities do not lead to consistent patterns. Figure 5.19 shows it in combination with feature 7 where the distinction between static and dynamic activities is quite reliable.

Attitude of the Sensor

The attitude of the sensor, feature 19, takes into account the signals shown in Equations (5.29) and (5.30).

$$f_{19}(\vec{E}) = att_{64}\left(\left|a_h^{BF}\right|, a_v^{BF}\right) = \left(\Delta\overline{\left|a_h^{BF}\right|}_{64}\right)^2 + \left(\Delta\overline{a_v^{BF}}_{64}\right)^2 \tag{5.52}$$

Δ refers here to the difference of the current value to the values during *standing* which have been saved in the calibration phase.

While all features in the body frame refer to the attitude of the sensor to some extent, feature 19 particularly exploits the attitude of the sensor. This feature gives information about the attitude difference between the current activity and the known sensor attitude for *standing*. As such, it is the main discriminator between *standing* and *sitting* – the most difficult decision for activity recognition. The difficulty is further illustrated in Figure 5.22.

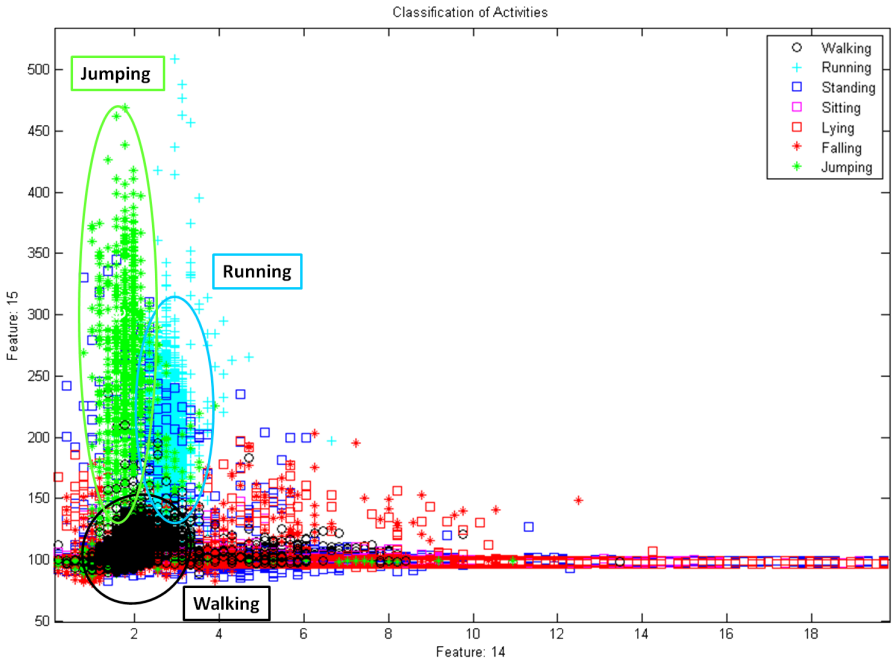


Figure 5.21: Plot of the main frequency component of $|a|$, $MFC_{128}(|a|)$, (Feature 14) and the energy of $|a|$ below 2.85 Hz, referenced as $\hat{E}_{128}\left(\overset{2.85\text{ Hz}}{\text{LPF}}(|a|)\right)$ (Feature 15) for all the activities. The main frequency component of $|a|$ during *walking* is around 2 Hz, while it lies around 3 Hz during *running*. This feature is very useful for distinguishing *jumping* and *running*, as even repeated *jumping* has a different frequency as *running*. A distinction between *walking* and other dynamic activities such as *jumping* and *running* is possible, as feature 15 measures clearly higher values, i.e. higher energy in its frequency band, for *jumping* and *running* than for *walking*.

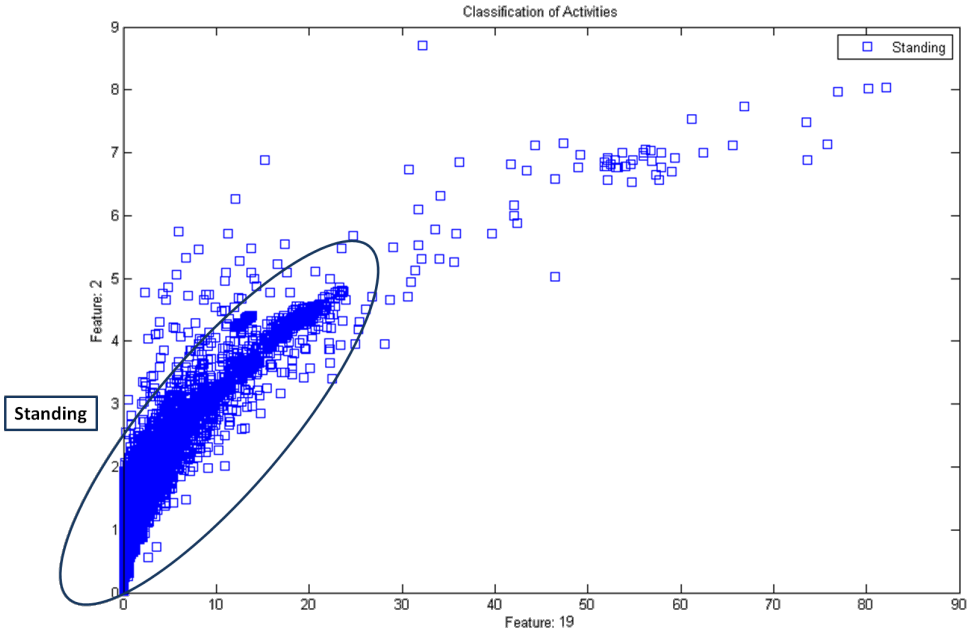
5.3.2.4 Feature Quantisation

The ranges of all features presented above are defined by continuous functions. To ease inference on resource constrained devices, they are discretised – a typical approach for this objective [66]. Along with the discretisation, the value range of each feature is meant to be partitioned into a minimum of intervals which have different impact on the physical activities to be inferred.

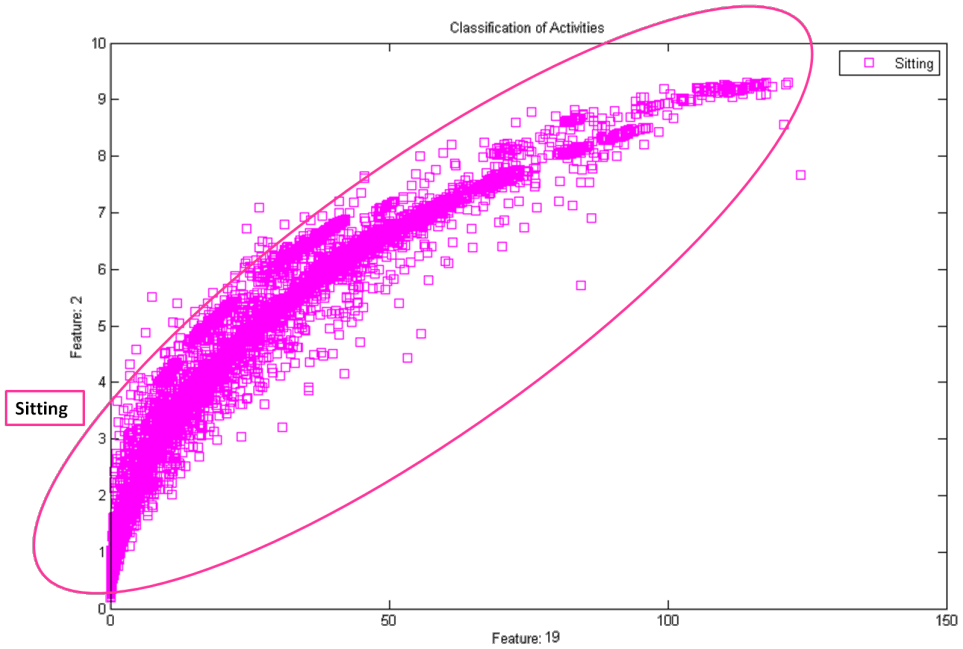
Automatic clustering processes can be used to quantise the features. Example clustering algorithms would be for instance the density based *k-Means* algorithm based on the Euclidean distance between observed values for a feature, which can be implemented efficiently [77], or again the Expectation Maximisation, see section 2.3.4.

When such algorithms are used for one feature, i.e. one dimension only, they can reasonably define the clusters and identify outliers, however without incorporating knowledge about the activity. Physical activity as a nominal feature (i.e. the values do not have a natural distance function defined between them) cannot be included guaranteeing good results.

Therefore the feature quantisation process has been manually realised in this work using histograms of the feature and an inspection of the caused activity discrimination in combination with other features, as shown in Figure 5.23. The quantised feature is the main frequency component of $|a|$, f_{14}. The physical activities mainly influenced by this feature are *walking, running, jumping* and *falling*. The histograms (Figure 5.23 a) and the

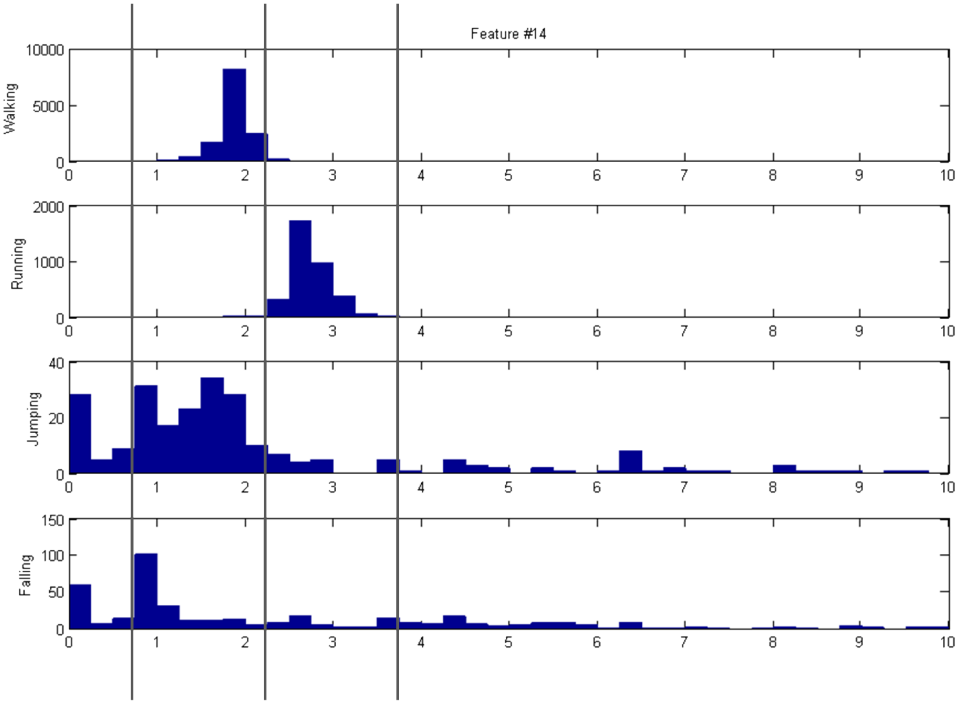


(a)

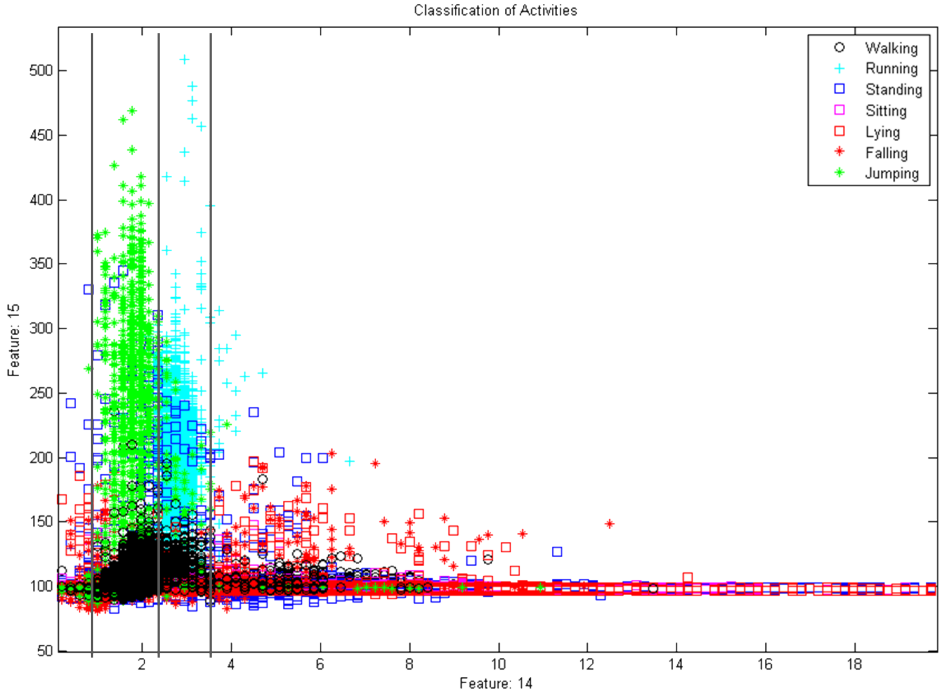


(b)

Figure 5.22: This figure shows the problem to distinguish between (a) *standing* and (b) *sitting* with the similarity of the behaviour for feature 19, $att_{64}\left(\left|a_h^{BF}\right|, a_v^{BF}\right)$ and feature 2, $\overline{|a_h^{BF}|}_{128}$. When the user is sitting in such a position that the body attitude (at the placement of the sensor) is similar to *standing*, both activities can be confused. Only incorporating a transition model between activities can improve the inference of both activities.



(a)



(b)

Figure 5.23: Example of the feature quantisation for the main frequency component of $|a|$, Feature 14. It mainly is relevant for *walking, running, jumping* and *falling*. Four states can be identified with the help of the histograms (a) and the plots of a pair of features (b).

two-dimensional plot of the feature (Figure 5.23 b) suggest four intervals of these states, the numerical values of which are determined with the help of the graphical representation.

5.3.3 Inference of Human Motion Related Activity

Inference has to decide which of the seven considered physical activities have effectively caused the measured values of the 19 features. This is a general classification problem, approached here with Bayesian techniques.

For convenience, this section abbreviates the *human motion related activity* or *physical activity* from time to time by *activity*. When necessary, the distinction is made to *high level activities* which can incorporate other information than inertia information.

Using the features identified in the preceding section as discrete random variables $F_i, 1 \leq i \leq 19$, Bayesian techniques classify new measurements as the different states of the random variable *Activity* using probabilistic inference mechanisms as described in section 2.3.

Therefore the Bayesian network with the random variables described above has to be learnt with the theory described in section 2.3.4. Basis for learning the Bayesian network is the data set presented in the following subsection. After that, the learnt static Bayesian network for both the full Bayesian and the naïve approach is explained and its extension to a hidden Markov model for dynamic Bayesian inference is shown.

To support the dynamic Bayesian approach, a further state for the RV *Activity* is introduced: *transition*. It reflects for instance *getting up* or *sitting down* – all transitions between the seven defined activities. No extra feature has been learnt for *transition*.

5.3.3.1 Data Set

To provide data for learning the Bayesian network, a labelled data set has been recorded. The objective is to create one Bayesian network which is usable for all persons, which is realistic as the selected features do not depend to a large extent on the physical constitution of the individuals. For labelling, this research has used the shadowing technique in which a human observer tracks start and stop time of each physical activity. The time periods between the end of the last activity and the start of the next one is automatically labelled as *transition*.

A total of 16 people, 6 females and 10 males aged between 23 and 50 years of different height, weight and constitution, participated in the acquisition of the test data set. They were asked to execute a sequence of activities in the given order, to allow to cover all activities and to ease labelling. Test candidates were asked to execute the tasks in their personal style without a strict choreography. They even were encouraged to perform the same activities differently and to perform these activities in any way that a human observer could identify.

Data were recorded in indoor and outdoor environment under semi-naturalistic conditions. The human observer was carrying a laptop computer to which the sensor was connected. This person was responsible for the labelling with a dedicated graphical application. The sensor was placed on the belt of the test candidate either on the right or the left part of the body. The data set comprises all different sensor positions. In order to check orientation performance of the sensor, test candidates performed their activities also with different headings.

Activity	Duration (minutes)
sitting	55
standing	107
walking	70
running	15
jumping	7
falling	2
lying	25
transition	4

Table 5.1: Constitution of the data set per human motion related activity.

The final data set contains more than four hours and thirty minutes of physical activities together with the sensor measurements. Table 5.1 shows the exact amount of recorded data per physical activity.

The features in the data set have been calculated with 4 Hz. The minimum duration of one considered physical activity is around one second for the short time activities *jumping* and *falling* which again consist of different phases. With four inferences per second of the current activity, no significant phase of any activity has been missed. More frequent inference should be neglected if this does not add significant value.

While it is desirable in theory to compute the features as often as possible, in order to keep the values always up to date and in addition to have a large data set of features for learning the Bayesian network, feature computation is a resource intensive process which should be reduced to the necessary minimum. Furthermore it is sensible to compute features only when an inference process uses them. Therefore 4 Hz have not been exceeded. The study in [237] moreover has shown that no significant change of inference quality can be achieved by increasing this frequency.

5.3.3.2 Static Bayesian Network

To infer the current motion related activities, static Bayesian networks can be used. The observed features with their activity labels have been used to learn the conditional probability tables and priors of all random variables. The structure can either be defined as a Naïve Bayes model, or can also be learnt, cf. section 2.3.4.4.

Naïve Bayes Structure:

The Naïve Bayes assumption presented in section 2.3.2 says that the random variable *Activity* is the single cause influencing all feature random variables $F_i, 1 \leq i \leq 19$. Like this, the structure is unambigously defined as shown in Figure 5.24.

While the BN structure is predefined, the conditional probability tables of the features and the prior of *Activity* can be either set manually or learnt from the data set with the techniques described in section 2.3.4 on page 39, in particular applying Eq. (2.15) assuming a Dirichlet distribution of the random variables.

While the automatic learning approach was used for the CPTs in this work, the priors have been set manually. The learnt priors would have reflected the relative frequencies of the data set visible in Table 5.1 and therefore overrate rare activities like *falling*, *jumping* and *running*. Even *standing* is overrepresented in the data set, as it is used for calibration in our approach. The static classifier uses instead the prior table given in 5.2.

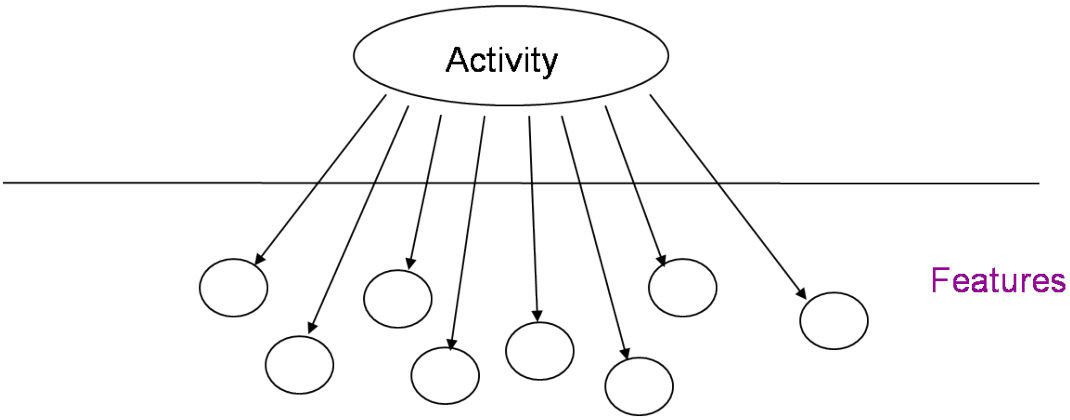


Figure 5.24: Naïve Bayes approach for activity recognition. The activity the user is performing is the cause of the observation of the features.

$activity_i$	*sitting*	*standing*	*walking*	*running*	*jumping*	*falling*	*lying*	*transition*
$P(Activity = activity_i)$	0.195	0.2435	0.409	0.001	0.001	0.0005	0.14	0.01

Table 5.2: Prior probabilities of the node *Activity* used for inference in the static Bayesian networks.

The Naïve Bayes assumptions of the independence of the features eases also inference of the current activity, i.e. the activity which maximises the following equation:

$$activity = \arg\max_i \left(P(activity_i|F_1, F_2, ..., F_{19})\right) \tag{5.53}$$

Applying Bayes' Theorem (Eq. (2.2)), the Naïve Bayes assumptions (Eq. (2.10)) and neglecting constant terms which do not influence the outcome of a $argmax$ expression, we can simplify inference as follows:

$$\begin{aligned} activity &= \arg\max_i \left(P(activity_i|F_1, F_2, ..., F_{19})\right) = \\ &= \arg\max_i \left(P(F_1, F_2, ..., F_{19}|activity_i)P(activity_i)\right) \\ &= \arg\max_i \left(\prod_{j=1}^{19} P(F_j|activity_i) \cdot P(activity_i)\right) \end{aligned} \tag{5.54}$$

Hence inference only has to infer the activity that maximises $\prod_{j=1}^{19} P(F_j|activity_i) \cdot P(activity_i)$ where $P(F_j|activity_i)$ are directly known from the evidence and $P(activity_i)$ is the defined prior probability (see Table 5.2).

As all information necessary for inference is directly available in the BN model, no specific inference algorithm like PPTC has to be used.

Learnt Bayesian Network Structure:

If the Naïve Bayes model is not assumed, also the network structure has to be learnt. As the random variables however are all known and the data set is complete, the simpler case of learning with complete data applies, as presented in section 2.3.4.2 on page 40.

This work has applied the *K2* algorithm of Cooper and Herskovits [71] using a Log score function to rate possible BN structures proposed by a Greedy Hill Climber with random restarts. Still some measures are taken to reduce the search space among the network structures:

- The upper bound for the number of parents per node is set to 5: this feature is directly offered by the *K2* algorithm.
- The Naïve Bayes model is set as the starting point of the *Greedy Hill Climber* algorithm, so every network candidate has to produce a better fit to the test data than the Naïve Bayes model.
- The directions of the edges in the Naïve Bayes model must not be reversed, can only be removed.

The learning process has shown that relations between the features are found. A hypothesis for this result is illustrated in Figure 5.25: the inter-feature relations are probably caused by unobserved nodes which are a common cause for some of the features. These unobserved nodes could represent different parts of the body and/or signals which influence more than one feature. As these nodes however are unobserved, the consideration of the dependencies of the features improves the score of the proposed network structures in the *K2* algorithm. I.e. they represent the data set better than the Naïve Bayes model and therefore lead to better inference results.

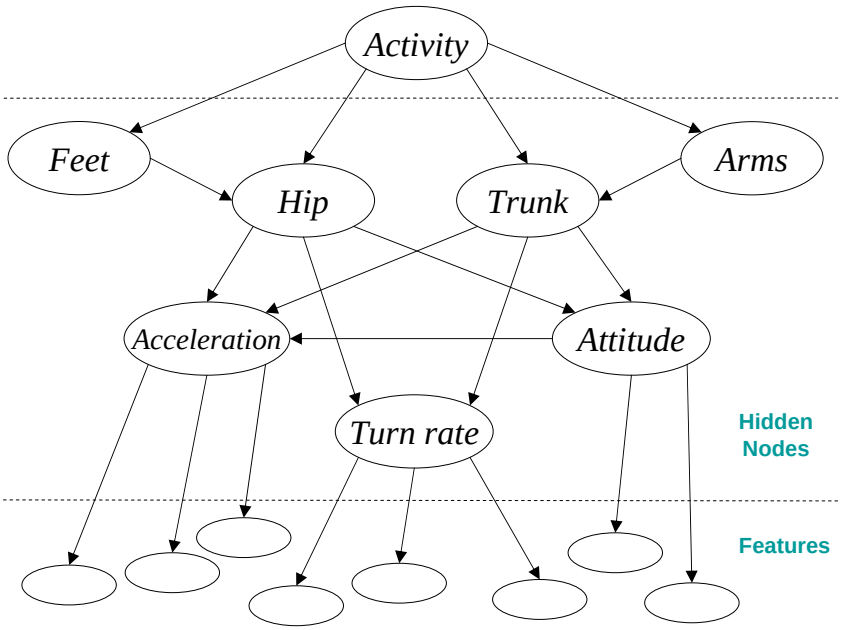


Figure 5.25: Hypothesis for the unrestricted Bayesian network approach. Consideration of the dependencies of the features is required because hidden nodes in the network due to the location of the sensor and the origin of the features exist.

The result of the learning process is a BN structure which represents a local maximum of the score function. As a random effect creates new network candidates, different learning processes can yield different results.

The resulting BN can be further minimised taking into account that all features are observed and only *Activity* is inferred. In this case, inference only has to include the Markov boundary (see section 2.3.2) of *Activity* which consists of all children of *Activity* and their parents which are not children of *Activity*. Note that with the imposed limitations for structures, there are no parents of *Activity*.

Knowing that also the (non *Activity*) parents of the children are observed, all incoming edges to them can be removed. Their parents have to be in the Markov boundary anyway (all other nodes are already removed) and the edge does not impact the value of *Activity*.

A network structure resulting from this learning and pruning process can be seen in Figure 5.26.

In general BNs with unknown structure require complex inference algorithms, such as PPTC (cf. section 2.3.3), which are not always suitable for real-time inference. But also inference with the learnt Bayesian network structure can be simplified. This is possible, as the required probability $P(Activity|\mathbf{O})$ with $\mathbf{O} = \{F_i, ..., F_{19}\}$ needs to be computed of a single RV whose Markov boundary carries evidence in all its nodes (see section 2.3.2 for details). The present case can guarantee evidence in all feature RVs, as their value is directly and locally calculated from the IMU, hence calculation is $P(Activity|\mathbf{o})$.

Combining the inference formulation from Eq. (5.53) and the factorisation of the JPD in BNs from Eq. (2.9) in the simplification process of Eq. (5.54), the following term has to be calculated:

$$activity = \arg\max_i (\prod_{j=1}^{19} P(F_j|pa(F_j)) \cdot P(activity_i)) \tag{5.55}$$

As all features carry evidence, $P(activity_i, F_1, F_2, ..., F_{19})$ can be calculated immediately by multiplying the correspondent values of the features' CPTs and priors for all possible values of *Activity*. This method can be used for real-time activity recognition as it consists only of a limited number of table look-ups and multiplications.

5.3.3.3 Dynamic Bayesian Network

Based on the static BNs presented above, the utility of *dynamic* BNs in comparison to static ones is investigated by defining a dynamic BN on top. It defines and quantifies discrete activity transitions and allows for evaluation with a *grid based* filter, see sections 2.3.5 and 2.3.6, pp. 44 et seqq., for the general theory.

In general, the last physical activity of a person influences his or her current activity. For instance, if somebody is currently *lying*, the most probable activity he or she will be performing immediately afterwards is *getting up* or still *lying*, but usually not *falling* and certainly not *running*.

Hence the knowledge about likely and also impossible sequences can provide valuable input for activity recognition. In terms of Bayesian networks, a new random variable carrying soft evidence is included as parent of the current physical activity $activity_t$ at time t.

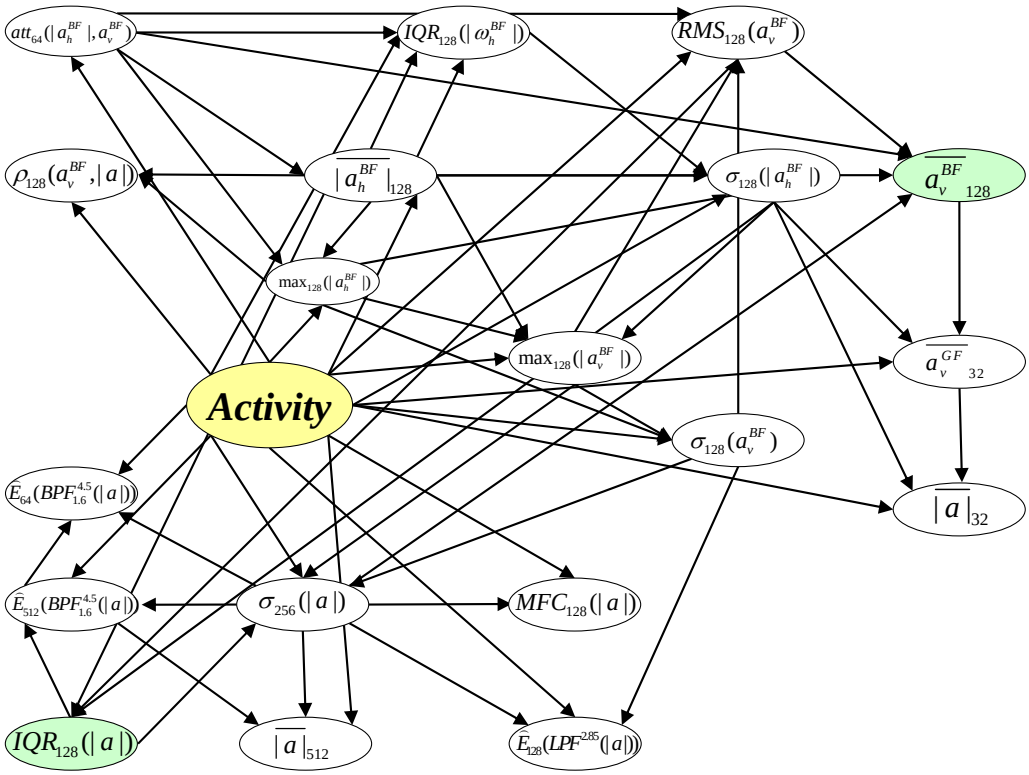
From this observation, as well as from the theory on filtering in section 2.3.6 it is easily understood that this extensions adds some complexity to inference. Section 6.2 on page 164 will evaluate, whether the qualitative improvement of recognition justifies the added complexity.

Figure 5.27 shows the HMM defined for the recognition of physical activities. $\mathbf{O}$ represents the set of all observed random variables $F_i, 1 \leq u \leq 19$ representing feature f_i. It is a first order HMM, hence the simplest DBN, but as all features are considered observed and only *Activity* is a *state variable* it is a suitable model. Higher orders of HMMs could model typical sequences like *sitting*, *gettingup*, *standing* which are however neglected here.

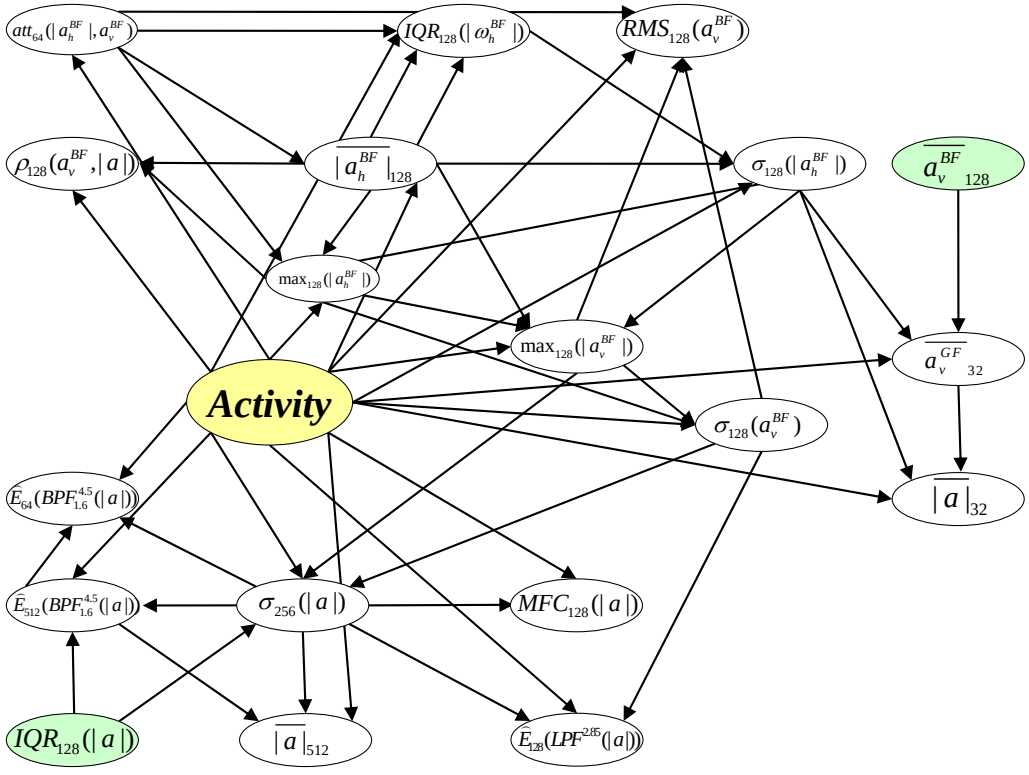
This first-order HMM for activity recognition is (following the definition in section 2.3.5) characterised by:

$$\lambda \sim (\mathbf{A}, \mathbf{B}, \pi) \ , \tag{5.56}$$

where:



(a) Structure of the learnt Bayesian network. All features are inside the Markov boundary of the node *Activity*. F_5 and F_{13} are not children of *Activity*



(b) Pruned version of the learnt network structure. Edges to F_5 (Eq. 5.38) and F_{13} (Eq. 5.46) can be removed, as they carry evidence and are not direct children of *Activity*.

Figure 5.26: The static Bayesian network used for the inference of human motion related activities. The pruned version (b) has less links and is therefore faster to compute while representing the same information.

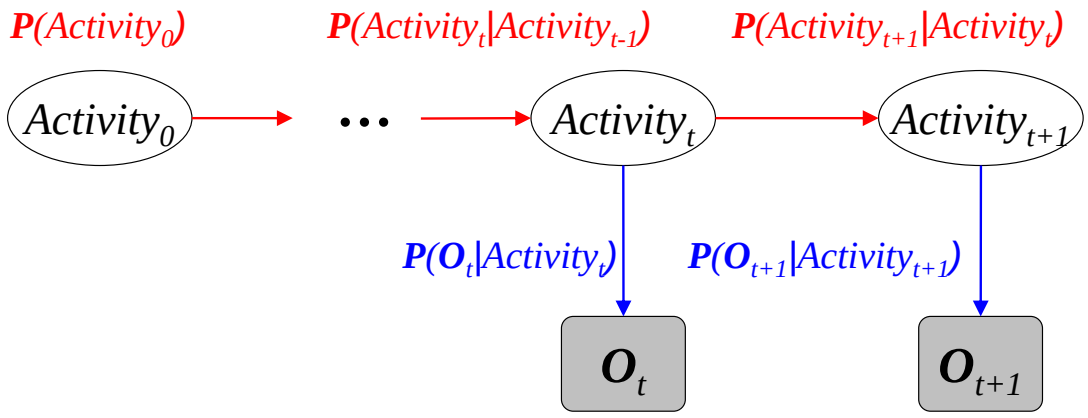


Figure 5.27: Hidden Markov Model for the inference of human motion related activities. $P(Activity_t|Activity_{t-1})$ has been manually configured by domain expert knowledge, $P(\mathbf{O}_t|activity_{i,t}, \lambda)$ are given by the underlying Bayesian network.

- Matrix $\mathbf{A} = \{a_{ij}|a_{ij} = P(acticity_{j,t+1}|activity_{i,t}), 1 \leq i, j \leq N\}$ is the state transition probability distribution or *transition model*. **A** has been defined by domain experts and is shown in Table 5.3. In this case $N = 8$ for the seven investigated and the transition activity.
- Matrix $\mathbf{B} = \{b_j|b_j = P(\mathbf{O}_t|activity_{j,t}), 1 \leq j \leq N\}$ is the observation symbol probability distribution in state j, defined by the measurement model in the underlying Bayesian network.
- **O** are the *observation symbols* which represent the observable physical or calculated output [65]. An observation symbol for a single point in time is in our case given by a vector with values for all features $f_1, f_2, ..., f_{19}$ computed from the raw sensor data. The vector of features is common for all the hidden states and can be denoted $O_t = (f_{1,t}, f_{2,t}, ..., f_{M,t})$, where $f_{i,t}$, $1 \leq i \leq 19$ is the value of the feature i at time t.
- $\pi = \{\pi_i\}$ is the initial state distribution, where $\pi_i = P(activity_{i,0})$ is the prior probability, $1 \leq i \leq N$ (see Table 5.2). As however the system starts in the calibration mode with the activity *standing*, evidence for *standing* is introduced in π.

	sitting	*standing*	*walking*	*running*	*jumping*	*falling*	*lying*	*transit.*
sitting	0.990902	0	0	0	0.000045	0.000045	0	0.009008
standing	0	0.566024	0.424518	0	0.004717	0.000024	0	0.004717
walking	0.033217	0.332171	0.498256	0.11626	0.003322	0.000166	0	0.016609
running	0	0	0.39984	0.439824	0.079968	0.0004	0	0.079968
jumping	0	0.36842	0.157894	0.263157	0.157894	0.000003	0	0.052631
falling	0.4	0	0	0	0	0.1	0.5	0
lying	0	0	0	0	0	0	0.818182	0.181818
transit.	0.666445	0.302929	0	0	0	0.000333	0	0.030293

Table 5.3: State transition probability matrix **A**. Each cell defines the transition probability $P(activity_{t+1}|activity_t)$ where the columns represent $activity_{t+1}$ and the rows $activity_t$.

The diagonal matrix **B** thereby can represent any Bayesian network. In particular, this work uses a $\mathbf{B}_1$ representing the Naïve Bayes network and a $\mathbf{B}_2$ representing the BN with the learnt structure.

The posterior of *Activity* can be inferred in this HMM with a grid based filter as described in section 2.3.6.1, as all involved random variables are discrete and finite. This approach provides exact inference for the clustered feature RVs. (Note that one could still call it approximate inference, as the features have been quantised.)

The estimation of $P(Activity_{t+1}|\mathbf{O}_{1:t+1}, \lambda)$ follows the general formulae for prediction and update in Equations (2.24) and (2.25) on page 46. The set of state variables $\mathbf{X}$, in the case of the HMM for activity recognition, consists only of *Activity*, the observed evidence $\mathbf{e_i}$ is represented by an instantiation $\mathbf{o}$ of the observed RVs $\mathbf{O}$.

Like this, the grid based filter can either calculate directly the prediction and update cycles by multiplications of the values in the conditional probability tables, or apply the matrix operations from Eq. (2.27):

$$\mathbf{P}(Activity_{t+1}|\mathbf{o}_{1:t+1}) = \alpha B A^T \mathbf{P}(Activity_t|\mathbf{o}_{1:t}) \tag{5.57}$$

Once the posterior probability distribution is estimated, the most probable activity can be given by the state with the maximum probability, if necessary.

5.3.4 Discussion

This section has presented a Bayesian approach for unobtrusive and resource efficient activity inference. The quality of its results will be evaluated in section 6.2 together with the comparison between the naïve and full, the static and the dynamic classifier.

All features have been selected for their physical and bio-mechanical sensibility. Also the modelling in Bayesian networks allows to interpret the results easily. With this approach, the same classifier will work also for persons whose data were not collected to train the Bayesian networks.

On the one hand, this effect cannot be achieved with pure unsupervised learning techniques. On the other hand, the automatic learning techniques from 2.3.4.2 ease the job a lot for the domain expert who do not have to specify a large quantity of conditional probabilities.

Using learning techniques for incomplete data sets could have identified hidden variables like those proposed in Figure 5.25, which would have resulted in a reduction of dependencies and a slimmer Bayesian network. The author however abstained from this option, as the fact that all nodes carry evidence but the queried *Activity* node, has reduced inference complexity enormously. The simplification in the BN structure could not keep up with this computation time reduction.

Most probably, the proposed set of features can be reduced without decrease in the inference quality. As can be seen in Table 5.4, every activity is covered by more than one feature. This redundancy however was used to add stability and to complete the characterization of the activities as opposed to pure recognition. A reason for this is the Bayesian approach and the reuse of the activity inference results in other inference modules. The higher the certainty of the activity inference approach, the better is the inference of higher level context in modules including the physical activity.

Also the extension of the present activity recognition system is eased with the redundancy in the features. As it characterises human motion, in general, it is expected that further activities like *climbing stairs* or *cycling* could be incorporated easily. The manually determined value ranges of the features will need respective adaptation, however.

Features	*Activity*						
Description	*standing*	*sitting*	*lying*	*walking*	*running*	*jumping*	*falling*
$f_1(\vec{E}) = \max_{128}\left(\left\lvert a_h^{BF}\right\rvert\right)$	×	✓	✓	×	✓	✓	✓
$f_2(\vec{E}) = \overline{\left\lvert a_h^{BF}\right\rvert}_{128}$	✓	✓	✓	×	×	×	×
$f_3(\vec{E}) = \sigma_{128}\left(\left\lvert a_h^{BF}\right\rvert\right)$	✓	✓	✓	×	×	×	×
$f_4(\vec{E}) = \max_{128}\left(a_v^{BF}\right)$	×	✓	✓	×	✓	✓	×
$f_5(\vec{E}) = \overline{a_v^{BF}}_{128}$	✓	✓	✓	×	×	×	×
$f_6(\vec{E}) = \sigma_{128}\left(a_v^{BF}\right)$	✓	✓	✓	✓	✓	✓	×
$f_7(\vec{E}) = RMS_{128}\left(a_v^{BF}\right)$	×	×	✓	×	×	×	×
$f_8(\vec{E}) = IQR_{128}\left(\left\lvert \omega_h^{BF}\right\rvert\right)$	×	×	×	×	×	×	✓
$f_9(\vec{E}) = \overline{a_v^{GF}}_{32}$	×	×	×	×	×	✓	×
$f_{10}(\vec{E}) = \overline{\lvert a\rvert}_{32}$	✓	✓	✓	×	×	✓	✓
$f_{11}(\vec{E}) = \overline{\lvert a\rvert}_{512}$	✓	✓	✓	✓	✓	×	×
$f_{12}(\vec{E}) = \sigma_{256}\left(\lvert a\rvert\right)$	✓	✓	✓	✓	✓	×	×
$f_{13}(\vec{E}) = IQR_{128}\left(\lvert a\rvert\right)$	✓	✓	✓	✓	✓	✓	×
$f_{14}(\vec{E}) = MFC_{128}\left(\lvert a\rvert\right)$	×	×	×	✓	✓	✓	×
$f_{15}(\vec{E}) = \hat{E}_{128}\left(\overset{2.85\ \text{Hz}}{\text{LPF}}\left(\lvert a\rvert\right)\right)$	✓	✓	✓	✓	✓	✓	×
$f_{16}(\vec{E}) = \hat{E}_{64}\left(\underset{1.6\ \text{Hz}}{\overset{4.5\ \text{Hz}}{\text{BPF}}}\left(\lvert a\rvert\right)\right)$	✓	✓	✓	×	×	✓	×
$f_{17}(\vec{E}) = \hat{E}_{512}\left(\underset{1.6\ \text{Hz}}{\overset{4.5\ \text{Hz}}{\text{BPF}}}\left(\lvert a\rvert\right)\right)$	✓	✓	✓	✓	✓	✓	×
$f_{18}(\vec{E}) = \rho_{128}\left(a_v^{BF}, \lvert a\rvert\right)$	×	×	×	✓	✓	×	×
$f_{19}(\vec{E}) = att_{64}\left(\left\lvert a_h^{BF}\right\rvert, a_v^{BF}\right)$	✓	✓	✓	×	×	×	×

Table 5.4: Evaluation of the features in function of the activities. The most significant features for every activity are specified.

5.4 Bayeslets: Making Context Inference Tractable

The powerful methods for Bayesian networks are a good starting point for realising context inference, although section 2.3.3.1 has shown that probabilistic inference is NP-hard and therefore in general computationally intractable.

This problem cannot be overcome in general. Instead suitable models have to be found that limit the computational requirements and minimise inference time. Section 3.3 has imposed further requirements to such models, in order to limit inference time and remote communication. This section now presents a method to realise the postulated modular and adaptive context inference rules, reducing inference resources at the same time.

To limit the resources necessary for probabilistic inference in a NP-hard problem, the single factors determining the complexity have to be limited. Section 2.3.3.1 has shown that the complexity depends primarily on the network structure.

Complexity in m-ary trees is $O(2q^2 + mq + 2q) = O(N)$, as $m < N$ depends on the number of nodes N in the BN where q is the maximum cardinality of value ranges in the BN. For singly connected BNs, the complexity is $O(N \cdot q^e) = O(N \cdot q^N)$, as the upper boundary e for the number of parents of any node in the BN depends on N [96]. Finally, for the general NP-hard case, where only exponential algorithms are known, the order is exponential in the size S of the largest cluster of the junction tree [85], $O(c^S) = O(c^{q^N})$ for some constant c, because the size of the cluster depends on the cross product of the value ranges in a cluster which consists at least of the node and its parents, hence depends on N.

Therefore, to reduce inference time, the three factors N, e and q have to be reduced to the minimum necessary. The latter thereby depends on the current situation and is user specific. All these factors shall be addressed in the following sections.

Locally computable inference modules are a core to tractable context inference as has been shown in the last sections for location (section 5.2) and human motion related activity (section 5.3). However, as not all context information is inferable with local data only (as it includes different users' context for instance or information from a different device) there must be ways to access remote information. To this end a concept to plug and jointly infer context inference rules has been developed.

The remainder of this section shall first present methods to reduce all three complexity factors *nodes* (section 5.4.1), *arcs* (section 5.4.2) and *value ranges* (section 5.4.3). All approaches thereby are designed to fulfil the requirements of decentralised, modular inference, and the adaptability to the current situation. Section 5.4.4 then shows how the relevant inference modules for an inference target can be selected and composed automatically. The inference in such a distributed, composed Bayeslet network is described finally in section 5.4.5, before the presented overall concept is discussed in section 5.4.6

5.4.1 Reducing the Number of Nodes: Segmentation of Inference Rules

Bayesian networks representing the full knowledge of all known influence to a certain fact can be immense. A very often cited example is the BN of the Computer-based Patient Case Study (CPCS) described by Pradhan et al. in [64]. This multiply connected network covering knowledge about internal medicine consists of 448 multi-valued nodes and 906 links. Real-time inference with fast response times is not possible with such a network.

Equally the situation of a user of a ubiquitous computing system can encompass a very large range of human activities, sensor readings, system usage information, as well as such

data relating to other people – all represented as random variables in a very large BN. Obviously it is impossible to include all such RVs in a representation used in inference, as the resulting BN would be too large to be processable.

The example network from Figure 5.28 for instance only includes certain context information relevant for the high level context *HighlevelActivity* and *Availability* consisting already of 81 nodes and over 130 edges. And this is not even all information relevant for it. The network would have to be extended to incorporate also information such as heart rate and other bio sensor readings, past activity, and the status of projects being worked on. Also similar aspects of the people in proximity and other important persons, such as family or colleagues, play a role and would have to be modelled.

Therefore there has to be the possibility that for each user a different, personal and individual version of each network has to be stored and used respectively.

5.4.1.1 Characteristics

A complete Bayesian network modelling all possible influence factors would be intractable, as the size of the Bayesian network is an important factor for its computational complexity; as discussed before, inference time grows exponentially with the number of nodes. Therefore this work proposes to segment all this information into thematic groups called *Bayeslets*. This word has been composed of "Bayes", as Bayeslets represent knowledge in the way of Bayesian networks, and "let" signalling (like in the term "Applet") that they are small, mobile and self-contained modules, executable wherever needed.

The advantages of this approach are manifold, proposing solutions to the requirements of section 3.3.

1. *Resource saving context inference:*
   Requirement (1) on page 58 postulates decentralised, modular inference, in order to be able to infer high level context within the user's smart space, while accessing information from remote devices with a minimum delay and network traffic. Bayeslets represent such a modular approach, are small and therefore easy to store, and allow for distributed evaluation of inference rules.

   Not only the distribution to several computers speeds up inference, but already the modularisation itself. Sequential inference of several modules is faster than inference of one BN with all nodes. Inference time only increases linearly with the number of Bayeslets instead of exponentially in a monolithic BN. If each one of n Bayeslets consists of N random variables and general BN inference complexity is approximated with $O(a^N)$, then the combination of all Bayeslets is inferred in $O(n \times a^N)$, while the monolithic approach in $O(a^{n \times N}) = O\left((a^N)^n\right)$. More details on inference in Bayeslets will be given in section 5.4.5. A practical evaluation of the impact of reducing the number of nodes can be found in section 6.5.

2. *Situation adaptive context inference:*
   Bayeslets add flexibility to a BN, as they are easily (un-)pluggable according to the dynamically changing requirements of the environment and their relevancy to an inference target. Like this, inference is situation adaptive. Only currently relevant information is used for inference, in agreement with Requirement (2). More details on situation adaptive composition of Bayeslets will be given in section 5.4.4.

3. *Personalisable context inference:*
   As Bayeslets are small representation units that can be stored without further burden on a mobile device, it costs no additional complexity, if every mobile device and

Figure 5.28: Bayesian network for the inference of the high level context *HighLevelActivity* and *Availability* with 81 random variables.

therefore its user would have their own, personalised version of that Bayeslet. Personalisation itself is context aware and relates to all factors of BNs, random variables, their dependencies and their value ranges. This realises the adaptivity postulated in Requirement (6). Details on adaptation to the current situation are shown in particular for value ranges in section 5.4.3.

4. *Coping with heterogeneous context inference mechanisms:*
Support for different characteristics of context as in Requirement (4) can also be realised with Bayeslets. Every Bayeslet can be evaluated with a different inference algorithm which is most suitable for its information, as long as it provides a probability distribution specifying the belief in its result.

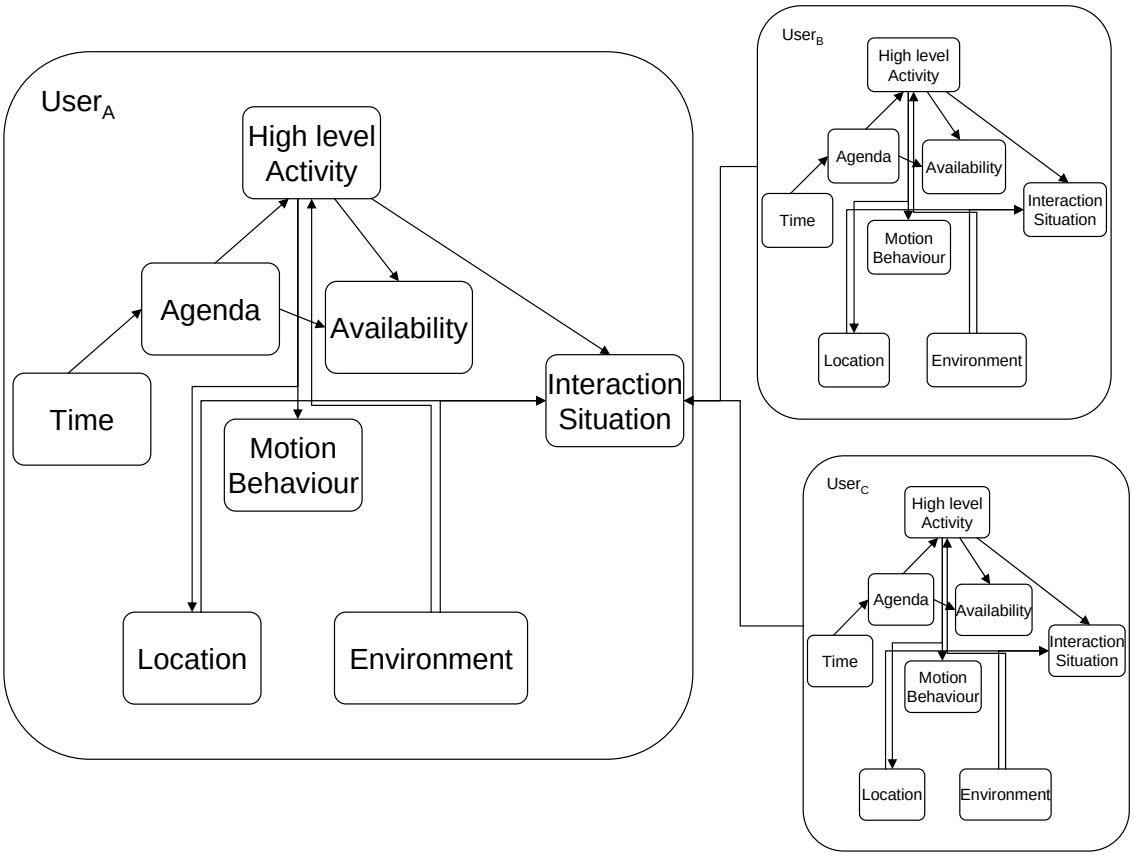


Figure 5.29: Inference network based on Bayeslets. The main components from Figure 5.28 are grouped into Bayeslets, shown in rectangles with rounded corners, which can be composed upon need.

Using Bayeslets as modular Bayesian networks, the BN shown in Figure 5.28 is grouped into several subdomains shown in Figure 5.29. Thereby, only those Bayeslets which are necessary for an inference request have to be connected, together with relevant information of other users which can differ from situation to situation.

5.4.1.2 Bayeslet Design

In Figure 5.29 it can be seen that location and motion activity of a user, the local and autonomous inference modules already explained in sections 5.2 and 5.3, are grouped into separate Bayeslets. Equally, all information related to other users is represented in different Bayeslets.

There are some general guidelines on how and where to separate all available random variables into Bayeslets, i.e. how to define the borders of domain knowledge.

1. Every sensor forms a separate Bayeslet together with the nodes influenced, as shown for the smallest case in Figure 5.30. Both, the number and the selection of sensors are not crucial for the inference target. It is desirable that sensors can be added or removed, that no sensor is used if not necessary and that several sensors can be used if it adds value. An example is shown in Figure 5.28 at the bottom right corner, where there are several sensors for the detection of the weather. Using Bayeslets, only currently relevant ones are selected and used for context inference.

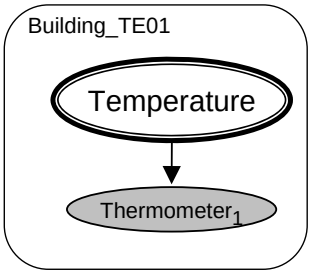


Figure 5.30: The most simple case of a Bayeslet describing a sensor and the quantity influencing it. In the example shown, it is the thermometer measuring the temperature of building TE01.

2. A Bayeslet only contains information about one user, the owner of the Bayeslet. Information about other users is plugged in, when necessary. The big advantage of this approach is that not all information of possibly relevant people has to be modelled and inferred upon, but only of the currently relevant people. While at home the family is most important and the status of line managers (depending on the job) usually can be ignored, this role is probably inverse during working hours.

   Furthermore the information letting infer other persons' context is usually not originating from the same device than the information about yourself. Separating this information avoids remote communication and facilitates local inference. Only remote inference results have to be transmitted, and only if the respective Bayeslet is used. Like this every user keeps control of his/her own information.

3. Unconnected parts of BNs can be separated into different Bayeslets. Obviously, if there is no (undirected) path between two random variables of a Bayesian network, they are independent and do not mutually impact on the inference of the other RV.

4. Parts of the overall information that would be d-separated by an evidence node in a monolithic BN can be divided into Bayeslets. If there is a node (set) A directly connected to a sensor or always available information, as it is specified for dynamic Bayesian networks and also for inference of motion related activities, it can be regarded as always observed. If A d-separates two parts of the network, these can be separated in different Bayeslets.

   A Bayeslet hence never needs to contain other nodes than those included in the Markov blanket (to be precise: the Markov boundary) of the currently inferred random variable. This suggests the explicit determination of nodes that can be inferred by a Bayeslet. The Markov boundary of these nodes define a hard border for the Bayeslet.

5. Very weak edges, i.e. such ones representing (almost) uniform distribution can be skipped, if this enables a separation. Removing edges can disconnect graphs, if the edge had been a *bridge*. A bridge is the only (undirected) connection between two subgraphs. After the removal, both subgraphs are obviously independent. Removing edges that have not been bridges, still can have influence on the d-separation of the formerly connected nodes and therefore make up the conditions of point 4.

6. A maximum number of nodes per Bayeslet can be defined. In the previous chapters we have seen Bayeslets for location or human motion activity. The latter includes about twenty random variables for one higher level RV and nineteen sensor RVs. With thirty nodes, also a little bit more complex Bayeslets with more than one related higher level RVs should be representable. Depending on the existing connections and number of observed nodes, also this size can be assumed to be computationally tractable on mobile devices.

   Cuts should be made by removing a minimum number of edges, hence preferably bridges, preferring weak links and such that enable d-separation. Furthermore, assuming evidence in nodes which are not in reality marked as evidence nodes, enables d-separation and can be used to force a separation into several Bayeslets.

Weakness of edges thereby can be determined with different means, e.g. Mutual Information of the linked random variables [105], or the deviation of the child's CPD from a uniform distribution.

In addition, it is fair to assume that human expert knowledge can be used to determine the RVs that make up a Bayeslet. Expert knowledge can also be used to propose Bayeslets which can be assembled to represent a problem domain such as *high level activity*. Expert knowledge is usually restricted to domains and therefore most appropriate to limit the representation of the domain to a core which is extensible by other Bayeslets.

In particular the combination of specifying network structures and learning the CPTs with techniques like those presented in section 2.3.4 is easy, time efficient and therefore realistic for practical use in ubiquitous computing. It has been applied e.g. for human activity recognition, cf. section 5.3.3.2.

5.4.1.3 Specification of Bayeslets

The preceding sections have presented the concept of Bayeslets and supported features. This section shall give an overview about how to specify a Bayeslet as a modular context inference rule usable by a CMS as sketched in section 5.1.

The main parts of a Bayeslet are the following:

1. Owning Entity
2. Output Context Attributes
3. Inference Algorithm
4. Input Nodes
5. Meta Information
6. Context Inference Rule

Factors (1) to (5) thereby constitute the *Bayeslet header* containing the general information necessary for the connection with other Bayeslets. The *Bayeslet body* contains the actual inference rule, evaluated by the inference algorithm.

Owning Entity:
The owning entity is the identifier of the Bayeslet's owner in the ubiquitous computing system. It is necessary, because a user might use different pseudonyms for different roles in the ubiquitous system, like for instance proposed in [15] or [11]. Different roles can have different inference rules depending on the purpose.

Output Context Attributes:
The output context attributes are the random variables inferable with a particular Bayeslet. In terms of DBNs, it is a subset of the state variables which, together with their Markov boundary, constitute the nodes contained in the Bayeslet.

A context attribute is defined by its name and the attribute's aspect. With the output attributes of a Bayeslet and the identifier of the owning entity, a Bayeslet can be identified. Identification is not unique, but does not have to be, either. Also several Bayeslets inferring the same output context (like sensors) can be plugged, if, as a result, inference quality can be increased.

Inference Algorithm:
To cope with the heterogeneity of data, not only Bayesian inference is allowed. The algorithm specified by this field needs to be available in the CMS and determines the interpretation of the Context Inference Rule field, as different algorithms require different inference models. Every type of classifier presented in section 2.2 can be specified here, in particular static Bayesian inference and the different Bayesian filters.

Input Context Attributes:
The input nodes specify the observed nodes in the inference rule, in terms of DBNs the evidence nodes. Only in these nodes other Bayeslets can be plugged. Only those Bayeslets can be plugged which provide the respective context attribute as output context attribute.

Output and input context attribute constitute the *interface* of a Bayeslet. It is visible in Figure 5.31 where input and output nodes are represented differently.

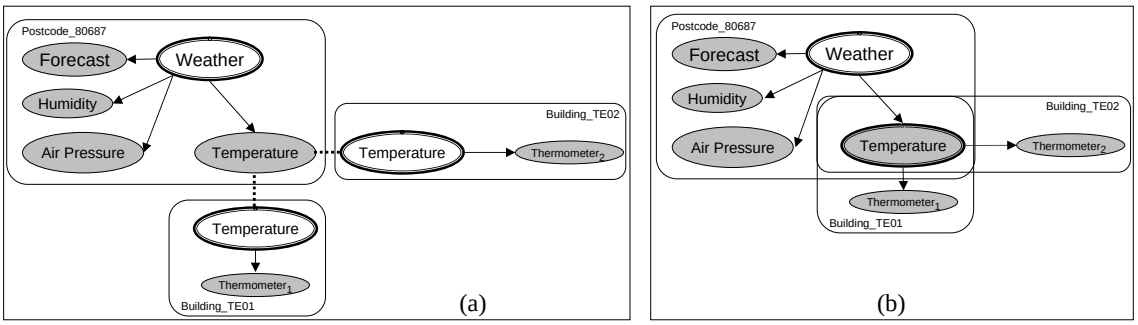


Figure 5.31: Connection of Bayeslets: the Bayeslet for the inference of the $Weather$ is connected to two Bayeslets determining the $Temperature$ incorporating a $Thermometer$. Input nodes are represented shaded in grey, output nodes have a double border.

Meta information:
A context inference system based on Bayeslets requires further information about the Bayeslets to increase its efficiency, as will be shown in the following sections 5.4.3 and 5.4.4. Examples for such meta information are the type of the value ranges (numerical, ordered, hierarchical, etc.), the expected posterior probability of the output nodes or the costs caused, depending on size, inference duration and inference frequency.

Context Inference Rule:
This field specifies the classifier evaluated by the inference algorithm, the actual inference rule module. Its contents depend on the inference algorithm and the corresponding model. For static discrete BNs it consists of the nodes with their states and the respective CPTs which are implicitly defining the BN structure. For Kalman filters it would consist of the normal distributions of the nodes and the matrices for transition and observation model.

For Bayeslets which are not evaluated locally, because they are related to remote information or information of a different user, no context inference rule is specified in this field, but a link to a remote access point providing the inference result.

Representation
Such Bayeslet specifications are exchanged between the different peers in the ubiquitous computing system, as a kind of inference service descriptions. Therefore they need a slim representation which is easily exchanged and fast interpreted.

XML is often used as an exchange format for service descriptions and could also be used here, as long as not too much overhead is produced. Easiest would be an extension of an available BN interchange format like PR-OWL [112], BNIF [54] or XMLBIF [55] with the respective header.

For use in a large scale ubiquitous computing system, the respective representation has to be standardised, this is however not the focus of the present research.

5.4.1.4 Summary

Bayeslets are the core building parts of an adaptive and tractable context inference approach for resource constrained devices. They are representing closed domains of knowledge (*encapsulation*) that is usable alone, but also in conjunction with further Bayeslets as shown in figure 5.29 (*modularity).*

The connection to further Bayeslets is enabled by predefined interfaces, specially tagged random variables called input and output nodes respectively. Several Bayeslets can be plugged into one input node. All pluggable Bayeslet candidates do not need to have the same input nodes, not even the same type of classifier, but only the output node demanded by this input node. This represents a kind of *polymorphism.*

Regarding input and output nodes as interface description for a class of Bayeslets, together with encapsulation, modularity and polymorphism, one can see the Bayeslet concept as an object oriented approach to probabilistic context inference.

The remainder of this work shall focus on Bayeslets representing discrete, static Bayesian networks as context inference rules like the example in Figure 5.31.

5.4.2 Reducing the Number of Edges: Personalised Bayesian Networks

The reduction of the number of arcs is important to reduce inference complexity, as it reduces the size of the CPTs which are the determining complexity factor for inference with the PPTC algorithm [85].

The reduction of edges is challenging, as they represent the dependence assumptions of the network. Every modification has a direct impact on the JPD of the BN. Hence reduction must not remove arbitrary links where CPTs would get too large. Instead the idea proposed here is to reduce the number of edges by personalisation of the network structures.

There are two stages to optimise the edges: when learning a network structure and when a structure already exists. In the following the usable approaches shall be sketched.

5.4.2.1 Learning Personalised Network Structures

In this section it is assumed that the scope of learning a BN is limited to a certain set of random variables constituting the Bayeslet and that the network is static. This eases structure learning slightly, assuming like in the preceding section maximum Bayeslets sizes between twenty and thirty nodes. This is still far too high to test all possible network structures, but can cover a more significant part than with more than eighty nodes as in Figure 5.28.

Using only individual usage history, only those dependencies will be learnt that hold for the user himself instead of all possible dependencies. My mood for example will not depend on the last Hurling results, whereas other persons are influenced strongly by them.

Structure learning algorithms also offer the option to limit the number of incoming edges per RV, like the *K2* algorithm of Cooper and Herskovits [71]. Also this approach reduces the learning space. Its drawback is the possible loss in quality of the network, as a broad range of structures are ignored. An advantage however is the reduced memory consumption during learning and the shorter inference time, which in the end overcomes the drawbacks, as we can also see in inference approximation (see section 2.3.3.3), where minor result divergences are accepted in turn for gains in computational efficiency.

Together with the limitations for the considered nodes and incoming edges, further knowledge can be included in structure learning:

- Causality between RVs (such as sensors readings being caused by a physical process) can be imposed, i.e. the presence and direction of arcs.
- A weaker form of the above is causality ordering, where the direction of an edge can be determined, in case a connection between two nodes exists. This limitation has been used for the creation of the BN for inference of human motion related activity in section 5.3.3.2.
- Arcs that are known to exist or to be missing between RVs can be specified, thus imposing dependence or independence between nodes or groups of nodes.
- Complete sub-units of the network that are assumed to be known can be defined.

Already in the last section, it was assumed that domain experts would have influence on the separation of Bayeslets. Using structure learning techniques they are not required to define the complete network structure, but only to specify the limitations like the above.

The domain expert can also be the user of the system. It is expected that the users can be shown graphical representations of their Bayeslets, in order to allow influence on the inference results. Addition or modification of links gives important information to limit the search space.

Together with the limitations of learning, a lower complexity class of the learnt BN structure can be targeted, i.e. tree-structured BNs or at least singly connected BNs with a fixed maximum number of parents per node.

There are two points where such structure learning modifications have to be included in the process described in section 2.3.4.4.

The first one is the greedy hill climber used to generate network candidates. It generates new network candidates by adding or inverting arcs in the last scored network candidate or by random restarts. It has to be ensured that the arcs specified by domain experts must not be modified and random restarts do not start at an empty network structure, but at the specified one.

The second point for modification is the scoring function rating the network structure candidates. They are already used to penalise the addition of arcs to avoid a fully

meshed BN structure [66]. To favour cycle free BN structures, those arcs can be penalised stronger which would introduce undirected cycles. For the detection of such arcs the same mechanisms can be employed that are used already to avoid the introduction of directed cycles.

5.4.2.2 Pruning Bayesian Network Structures

When a network structure is already learnt, the parameters, i.e. the conditional probabilities given the existing parents can be learnt from individual data. These personalised CPTs together with the specification of the Bayeslet can be used then to reduce the number of edges.

There is information in the Bayeslet specification which helps to prune edges: the knowledge about output nodes and input nodes. All (non-output) nodes of the Bayeslets are in the output nodes' Markov boundary, but not necessarily in the Markov boundary of every single output node. The input nodes are all assumed to carry evidence.

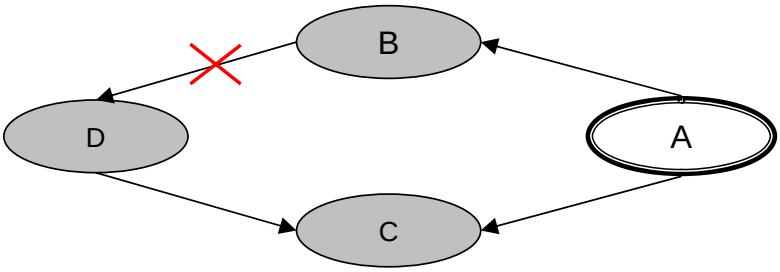


Figure 5.32: An example where a link between two nodes B and D in the Markov boundary of A can be removed to reduce the size of the conditional probability tables.

As described already in section 5.3.3.2, this knowledge allows to prune dependencies between nodes in the Markov boundary which do not have any influence on the requested output node. An example for such a situation is shown in Figure 5.32. As long as the Markov boundary is not changed,

- incoming edges of ancestors that carry evidence can be pruned.
- incoming edges to observed parents of siblings can be pruned.

In both cases, although also the origins of the pruned edges are within the Markov boundary, such edges cannot transport more information. The evidence of both nodes defining the edge to be pruned is already included via the links that have included them in the Markov boundary and, as only incoming edges are pruned, no *explaining away* effect [66] is omitted.

Furthermore, the information of the personalised conditional probability distributions can be used to detect edges which do not add a lot of value and could therefore be removed. Two possible approaches for determining the value of an edge are the degree of deviation from a uniform distribution and the entropy of the represented conditional probability.

Deviation from Uniform Distribution:

A uniform distribution of the conditional probability $\mathbf{P}(X|Y)$ represents no influence of Y on X. The larger the difference ΔU hence, the more important is the edge. In order

not to depend on the size of the value ranges of X and Y which are denoted q_X and q_Y respectively, the result is normalised.

$$\Delta U(\mathbf{P}(X|Y)) = \sum_{j=1}^{q_Y} \frac{\sum_{i=1}^{q_X} \frac{\left|P(x_i|y_j) - \frac{1}{q_X}\right|}{q_X}}{q_Y} = \frac{1}{q_X q_Y} \sum_{j=1}^{q_Y} \sum_{i=1}^{q_X} \left|P(x_i|y_j) - \frac{1}{q_X}\right| \tag{5.58}$$

Conditional Entropy:
In information theory, the conditional entropy $H(X|Y)$ is a measure for the information remaining in X when Y is known. This entropy $H(X|Y) = 0 \Leftrightarrow X$ is independent from Y. It is defined as follows:

$$H(X|Y) \equiv -\sum_{j=1}^{q_Y} P(y_j) \sum_{i=1}^{q_X} P(x_i|y_j) \log P(x_i|y_j) = -\sum_{j=1}^{q_Y} \sum_{i=1}^{q_X} P(y_j) P(x_i|y_j) \log P(x_i|y_j) \tag{5.59}$$

Note that the conditional entropy is not normalised and increases with higher values for q_X and q_Y.

Both approaches can be used to reduce large conditional probability tables. The closer to 0 both measures are, the less important is the link. Pruning the link to the parent with the largest value range yields the largest performance gain.

5.4.3 Reducing the Number of Values: Dynamic Value Ranges

Once the size of the BN or Bayeslet under evaluation is limited and also the number of parents is bound with the techniques described in the last sections, the number of values of the random variables should be limited as well. Again, it is important to reduce inference time while not loosing important information.

This section discusses the dynamic reduction of value ranges (VR) in already constructed, discrete Bayesian networks. The proposed methodology follows a bottom-up approach where initially each variable begins with its complete VR which may contain a high number of values. Subsequently, the number of values may be reduced (recursively if necessary) by merging values.

The following paragraphs shall first describe the algorithm and the modification of the conditional probabilities during reduction. Then different criteria are presented to select the values to be reduced and the criterion when values should be merged or re-expanded. The description follows the publication of the author et al. in [119].

5.4.3.1 Value Range Repartitioning Process

An example for the application of *dynamic value ranges* is the usage of symbolic location information (see section 5.2) for high level context inference. While at work, the room in which a user is located gives important information about his current high level activity. The basement garage implies coming or leaving, the meeting room a conference, not to mention kitchen or office. The number of rooms the user knows not only at work, but also at home, at friends' places and so forth is immense and not realistically usable.

For use in context inference hence, the VR of the random variable *Room* has to be reduced to the currently relevant ones. Therefore those rooms that are currently not relevant for not in reach can be merged into one value *other* of the RV *Room*. When the user leaves his workplace and heads home, the value range has to be modified again - the

values concerning travelling and home have to be expanded from the *other* state, while the work values can be merged.

This process of grouping values into partitions of the full value range and dynamically modifying the partitions upon need is called *repartitioning*. Repartitioning uses different methods and criteria to minimise information loss and computational cost of the repartition process. The process for the dynamic reduction of the value ranges is shown in the flow diagram of Figure 5.33.

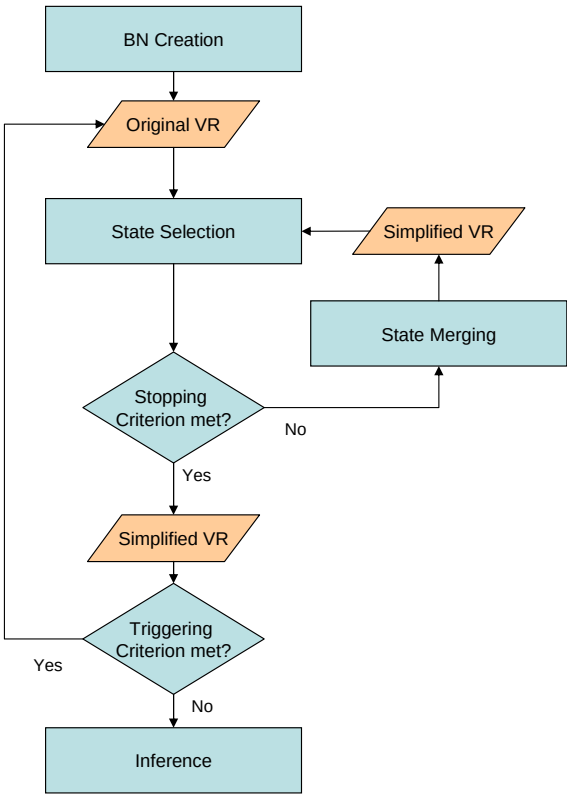


Figure 5.33: Flow diagram of a system allowing for dynamic repartitioning of value ranges of random variables.

State Selection selects values (also called states) to be merged with each other based on the selected selection criteria. State selection continues to select values for merging recursively until a stopping criterion is met. Merging includes node-internal changes (more or less an addition of the likelihoods to be merged) and updates of the children where the respective columns of the CPTs have to be merged. Once a VR is simplified, the dynamic repartitioning system still has to monitor changing conditions like context or current evidence, which might trigger a new repartitioning process for which the states have to be separated again, starting at the original value ranges.

5.4.3.2 State Merging and Separation

Dynamic repartitioning implies modifications in the conditional probability tables of the nodes. Merging a set of values causes the following changes:

1. *Node Internal modifications*:
   Merging a set of values only impacts the conditional probabilities of the merged

values, as the values of a RV are mutually exclusive and exhaustive. The conditional probability of a new *super-state* $\mathbf{v_m} = \{v_{m_0} \cup v_{m_1} \cup ... \cup v_{m_r}\}$ of RV V given some configuration of parents states $\mathbf{Pa}(V) = \mathbf{pa}_j(V)$, is :

$$P(V = \mathbf{v_m}|\mathbf{pa}_j) = \sum_{v_{m_i} \in \mathbf{v_m}} P(v_{m_i}|\mathbf{pa}_j) \tag{5.60}$$

2. *Update of Children's CPTs*:
After the modification in the value ranges of a node V, it is necessary to update the CPTs of its child nodes where the new parent configuration $\mathbf{pa}'_j$ contains the merged state $\mathbf{v_m}$. Again, taking into account that all states of a RV are disjoint, the new conditional probabilities of a child node $Y \in \mathbf{Ch}(V)$ can be computed as:

$$P(Y = y|\mathbf{pa}'_j(Y)) = \frac{\sum_{v_{m_i} \in \mathbf{v_m}} P(y, v_{m_i}, \widehat{\pi})}{\sum_{v_{m_i} \in \mathbf{v_m}} P(v_{m_i}, \widehat{\pi})} = \alpha \sum_{v_{m_i} \in \mathbf{v_m}} P(y, v_{m_i}, \widehat{\pi}) \ , \tag{5.61}$$

where $\widehat{\pi} = \mathbf{pa}'_j(Y) \setminus \mathbf{v_m}$ and α is the normalising constant.

The term $P(v_{m_i}, \widehat{\pi})$ is the joint probability of a configuration $\widehat{\pi}$ with every single state of V that was merged in the previous phase. The calculation of this term depends on the relation between the parents of Y and could be costly. It is however only normalising the new conditional probabilities to sum to one and therefore can be calculated ex post from the sum of conditional probabilities for Y and is replaced therefore by the normalising constant α.

$P(y|v_{m_i}, \widehat{\pi})$ does not add significant calculation costs as it represents the previous conditional probabilities $P(y|\mathbf{pa}_j(Y))$ of the original CPT.

Separation of states is more difficult. The (conditional) probability of a merged value $\mathbf{v_m}$ cannot be split into the respective probabilities for all $v_{m_i} \in \mathbf{v_m}$ and could only be approximated by a standard distribution, for instance the uniform distribution. As the original status however is stored in the Bayeslet specification, a separation of states just uses the original values for both, V and $\mathbf{Ch}(V)$.

The Bayeslet implementation therefore always has to maintain a list of merged states which is necessary for the separation, but also later on for evidence. If an original state v_{m_i} receives hard evidence, this has to be applied for $\mathbf{v_m}$ for inference.

5.4.3.3 Criteria for State Merging

As shown in the flow diagram from Figure 5.33, state merging is a recursive process selecting the most suitable states to merge until a final state is reached. The applied rules are called *state selection criteria*, the *stopping point criterion* decides whether the new set of values must be combined further or if the repartitioning process should stop.

There are a number of possible selection criteria for the most appropriate values $v_i, 0 \leq i \leq n$, in value range $\mathbf{R}_V$ of node V, the most relevant ones shall be presented in the following paragraphs:

Equal Number of States

The *Equal Number of States (ENS)* approach [98] divides the VR into subsets that take each m adjacent values and then merges all states of each subset. This method however

does not take into account the probabilistic information of the network. It could be useful for ordered, hence in particular numerical value ranges as of a RV *Temperature*, in order to reduce precision and therefore overhead.

Temperature	Degrees						
original	1	2	3	4	5	6	...
ENS processed	1-2		3-4		5-6		...

The quality of this merging depends largely on the probabilistic effects of the values to be merged. If they cause different behaviour in a connected node, the loss of information by merging would be quite high.

Tree Collapsing

If value ranges are not ordered, but define a tree of which the leaves are all possible values, another general approach is applicable. If the sum of all probabilities of values of the same branch are below a certain threshold, they can be merged to the branch's root node.

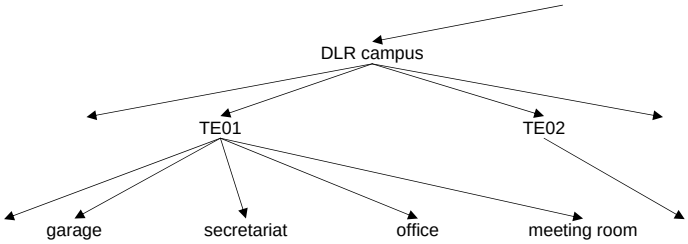


Figure 5.34: Part of a location hierarchy with the levels of districts, buildings, and rooms.

This is in particular applicable for hierarchically organised locations as they were used in section 5.2.4.2 where a building contains several rooms. If the complete building is not relevant, all rooms can be merged and represented by the building name. With regards to Figure 5.34, if the specified rooms on the lowest level are not relevant, they can be merged to the value on the building level: *TE01*.

Entropy Minimisation Discretisation

Entropy Minimisation Discretisation (EMD) adapts the method of [99], using Fayyad's definition of *class entropy Ent* for any value range subset $\mathbf{R}_{V,i}$ of $\mathbf{R}_V$:

$$Ent(\mathbf{R}_{V,i}) = -\sum_{v \in \mathbf{R}_{V,i}} P(V = v) \log(P(V = v)) \tag{5.62}$$

Based on this, he defined the *class information entropy induced by a binary partition.* Adapted to the partitioning of a value range $\mathbf{R}_V$ into two subsets $\mathbf{R}_{V,1}$ and $\mathbf{R}_{V,2}$ it can be defined as:

$$Ent(\mathbf{R}_{V,1}, \mathbf{R}_{V,2}) = \frac{|\mathbf{R}_{V,1}|}{|\mathbf{R}_V|} Ent(\mathbf{R}_{V,1}) + \frac{|\mathbf{R}_{V,2}|}{|\mathbf{R}_V|} Ent(\mathbf{R}_{V,2}) \tag{5.63}$$

The optimal partition is determined selecting the subsets $\mathbf{R}_{V,1}$ and $\mathbf{R}_{V,2}$ with the minimal $Ent(\mathbf{R}_{V,1}, \mathbf{R}_{V,2})$ amongst all the candidates. Merging all the states of each subset provides a binary discretisation.

Minimum Merged Probability

Minimum Merged Probability (MMP) tries to minimise the loss of entropy caused by repartitioning and is based on the concepts presented by Clarke and Burton in [97] for continuous random variables.

With the entropy, as already used in Eq. (5.59) above, it is possible to prove the following: If the states v_i and v_{i+1} selected to be merged are chosen in a way that the merged probability, $P(v_i) + P(v_{i+1})$, i.e. the probability of the resulting super-state, is minimised, the reduction of the entropy is minimised as well. Therefore the states with the minimum merged probability will be selected.

As the maximum entropy occurs with the maximum number of values, the best trade-off between maximum information and a manageable number of values is reached when the change in the size k of the value range becomes greater than the change in the entropy of V. This point is called the *knee point* of the entropy function over the number of values. With these properties, it can be used as *stopping criterion* for the recursive merging process.

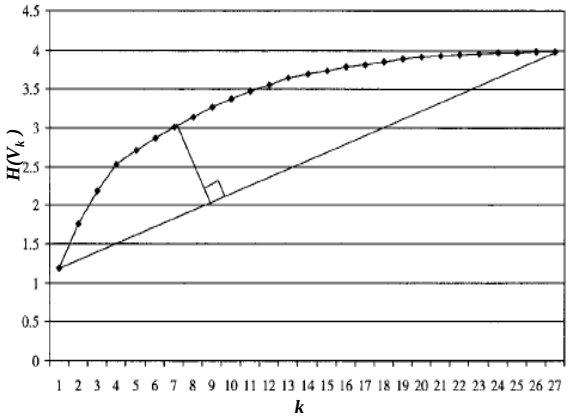


Figure 5.35: The knee point of an entropy function [97].

This knee point is the point with the maximum height over the straight line connecting $(k_{min}, H(V_{k_{min}}))$ and $(k_{max}, H(V_{k_{max}}))$, see Figure 5.35. The maximum height of a point on the function curve above this line is proportional to:

$$l(k) = k_{max} \cdot H(V_k) \; - \; k \cdot H(V_{k_{max}}), \tag{5.64}$$

k being the current number of states, $k_{max} = n$ the initial size of the value range and $H(V_k)$ the entropy of V with k states. Therefore, the stopping point is reached just before the decrease in this score [97].

An advantage of this method is the possibility to allow not only merging of adjacent values, but also of non-adjacent values. Processing only adjacent values is indispensable for the case of numerical or certain categorical values. For other value ranges however, selecting and processing the most suitable states regardless of their relative position within the VR, would give a better repartitioning result.

States of Interest

Very often the information necessary for the services requesting inference is contained in only few states of nodes of the network which shall be called *states of interest*. If the ubiquitous computing system has this information, it can apply the repartitioning methods only to the other nodes, in order to decrease the computational costs. The states of interest are *protected*, to maintain the information requested explicitly.

The states of interest can be fixed by expert knowledge at creation time of the Bayeslet or be determined dynamically by the requesting services as in Figure 5.36. Two different services require probabilistic information from an inference engine (included in a context management system) indicating their different states of interest of some context attribute. The inference engine can then generate two new Bayeslets from the original one. These Bayeslets will have different VRs in order to provide the inferred information with maximum performance and minimum loss of accuracy in the range of interest of each service.

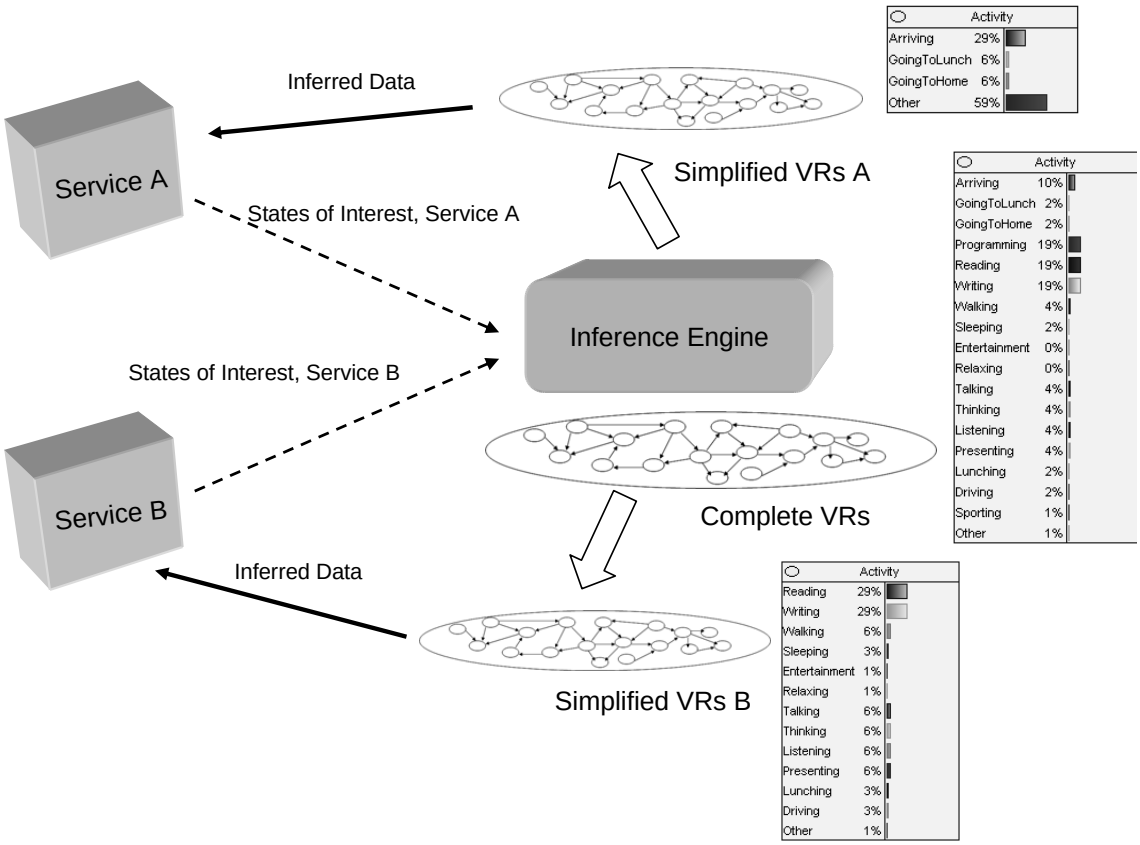


Figure 5.36: Model for dynamic value ranges where services can customise the inference process by specifying *states of interest*. Two services A and B request inference of context attributes together with a specification of their ranges of interest. The inference engine therefore simplifies the VRs for each service according to its needs.

Protection Extension

The use of any of the discretisation methods provides reduced quality for inference, i.e. increased variance with respect to the posterior inferred in the original system, even in the set of protected values.

In order to increase the quality, not only specification of states of interest is necessary, but also the extension of their protection to the most related values of other nodes. To allow for that, the values of the related nodes, i.e. parents and children nodes, have to be identified which are most relevant for the states of interest. This idea is realised by the *protection extension (PE)* approach.

Therefore, inspired by Shannon's *mutual information*[105], the following function is defined: the *partial mutual information* between the states of interest and the states of

the connected nodes represents a measure for the mutual relevance of states. For a set of interest $\mathbf{R}_{V,i}$ defining a super-state $\mathbf{v_m}$ for the node V and one value of its parent $X = x$, the partial mutual information is calculated as follows:

$$I(V = \mathbf{v_m}, X = x) = \sum_{v_{m_i} \in \mathbf{v_m}} P(x, v_{m_i}) \left| log(\frac{P(x, v_{m_i})}{P(x)P(v_{m_i})}) \right| \tag{5.65}$$

The absolute value is used to maintain the non-negative characteristic of the function, since the characteristic of $H(X, Y)$ is lost, as only a subset of the states of X and V is considered. Hence, the term $\left| log(\frac{p(x,y)}{p(x)p(y)}) \right|$ provides information about how dependent the states are. $P(x, v_{m_i})$ weights the expression based on the joint probability of the two states. Also other weighted measures could be adopted [100].

For the mutual information between a child's state $Y = y$ and a set of states of interest of its parent's state $\mathbf{v_m}$, the equation is:

$$I(Y = y, V = \mathbf{v_m}) = \sum_{v_{m_i} \in \mathbf{v_m}} P(v_{m_i}, y) \left| log(\frac{P(v_{m_i}, y)}{P(v_{m_i})P(y)}) \right| \tag{5.66}$$

Defining a boundary b, a minimum for the mutual information, the protected states can be chosen automatically. If $I > b$ the state is protected, otherwise not. If b is chosen small, more states will be protected, which leads to higher precision in the inferences at the cost of less reduction in the number of network states. For the remaining, unprotected states, any of the previous repartitioning methods can be applied, in an extreme case, all unprotected states could be merged into one super-state.

Summarising, if there are states of interest for some node, it is possible to calculate this measure for each state of the nodes related and extend the protection to those with the highest mutual information which in turn constitute a set of states of interest. If the protection extension is applied recursively to all connected nodes, it could be extended to all the Bayeslet or only a certain number of steps to save extensive calculations of partial mutual information.

5.4.3.4 Triggering Value Range Operations

As explained in the example for symbolic location above, a specific partitioning of a RV's value range does not have to hold for all circumstances. Therefore methods to trigger repartitioning during system run-time have to be provided, so the simplified value ranges can be adapted permanently to the latest requirements.

Therefore merged states also have to be separated again. As has been seen above, in order not to lose important information, the original state of the Bayeslet has to be stored.

The basic idea for a decision system that triggers repartitioning is the following: states are merged to a super-state, if they are not very relevant. If the probability of a superstate then exceeds a certain threshold probability, it is worthwhile detailing it by re-separating it into its original states.

It has to be taken care, however, that repartitioning is not called too frequently and that a single hard evidence in a super-state immediately causes repartitioning. Therefore a *filtered probability distribution* $\mathbf{P}_f(V)_t$ for the current VR of a node V at time t is calculated which keeps track of the last posterior distributions.

$$\mathbf{P}_f(V)_t = a \cdot \mathbf{P}_f(V)_{t-1} + (1 - a) \cdot \mathbf{P}(V|\mathbf{e}), \tag{5.67}$$

where $a \in [0;1]$ is a constant, $\mathbf{P}_f(V)_{t-1}$ is the filtered probability after the last posterior computation and $\mathbf{P}(V|\mathbf{e})$ is the posterior probability for V given evidence $\mathbf{e}$. $\mathbf{P}_f(V)_t$ in the present case of discrete RVs is a vector of filtered probabilities $P_f(V = v)_t \forall v$.

This filter preserves the characteristics of a probability distribution. Therefore it can not only be used as trigger for new repartitioning, but also in the state selection criteria (see section 5.4.3.3) instead of prior probabilities.

A threshold probability b for merged states can be set in relation to the size of the original VR. If $P_f(V = \mathbf{v_m}) > b$ by inference based on new evidence, a new repartitioning process is started.

5.4.4 Composition of Bayeslets

The application of the Bayeslet concept requires the introduction of an additional step before the actual context inference process: the assembly of the to be evaluated BN from single Bayeslets. In order to allow for efficient and precise Bayesian context inference taking into account the available information at request time, the concept of *Dynamic Composition of Bayeslets* is introduced by the author et al. in [122, 123]. As Bayeslets only represent knowledge domains which may be influenced by each other, for context inference, the related Bayeslets have to be selected and composed to represent all the relevant knowledge. As this relevancy depends on the current situation and the requested output node, composition has to be a dynamic process at run-time of the context aware framework.

Prerequisite for such information based assembly is a database of the available Bayeslets with the necessary information to compute costs and information gain. In this database all local Bayeslets are stored. Information about remote Bayeslets gets into the database by advertisements when remote smart spaces get in contact. The advertisements are not fully fledged Bayeslets, but only stubs representing the Bayeslet header and an access point for remote evaluation.

For joining two Bayeslets $\mathbf{X}$ and $\mathbf{Y}$, mechanisms to find pluggable Bayeslet candidates in the database are necessary. Plugging candidates are such Bayeslets where an input node of $\mathbf{X}$ matches an output node of $\mathbf{Y}$. Bayeslets with the same output node for different users can be distinguished by their identifier. When requesting a different user's Bayeslet via the specified access point, his security agent can check if the requester is authorised to obtain the information.

5.4.4.1 Dynamic Bayeslet Assembly Process

As described in section 5.4.1, the advantage of Bayeslets is the reduced size compared to a complete BN. So the inference rule composed of Bayeslets has to be minimal, but not neglecting important links. Moreover, assembly of Bayeslets poses challenges, as it is recursive, because linked-in Bayeslets can allow or require input from other Bayeslets. This bears two main risks:

- Loops: If a Bayeslet Network, i.e. the directed graph consisting of Bayeslets and their directed connections, would result in having a loop, also the complete BN formed by this graph could contain a directed loop, which is forbidden in BNs. Algorithms for deadlock prevention or e.g. breadth-first search can be applied to exclude such cases.

- Excessive Linking: if all available interface nodes in Bayeslets are linked and joint with other Bayeslets, at least in a large scale pervasive computing framework this

would result in a very large network, possibly until all Bayeslets in the knowledge base are linked. So the same, complete inference BN that was referred to in section 5.4.1 would be recreated. Its evaluation and already the time for joining and waiting for the output of the input's evaluation could lead to unacceptable high delays.

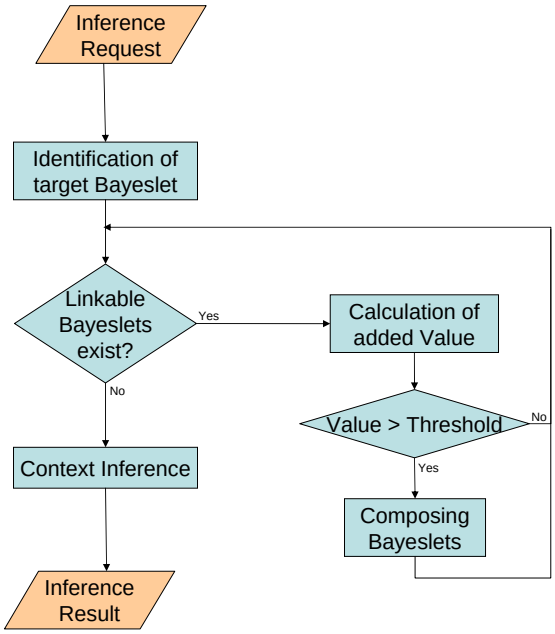


Figure 5.37: The process of composing bayeslets to a complete inference rule.

To answer an inference request, the Bayeslet with the target (output) node is selected and has to be assembled with Bayeslets linked into the input nodes. The selection of such Bayeslets has to be performed in a new step, preceding the inference itself as shown in Figure 5.37. Decision criterion has to be the costs added by joining the network weighted against the information gain. Costs are mainly the evaluation costs of a Bayeslet depending on the size and the communication costs, the information gain is determined by the expected impact on the posterior probability of the output node. Only those Bayeslets exceeding a certain threshold of added value would be joined.

5.4.4.2 Composition Criteria

To decide whether to connect two (or more) Bayeslets, the utility of the additional information has to be determined. In utility theory the term *utility* is defined by the value a piece of information provides to the system. Evidently, this value strongly depends on the usage of the information.

The utility of an additional piece of evidence $y \in Y$ can be quantified by the difference between the utility with the additional evidence and without the additional evidence. Here, y refers to the evidence of a random variable (e.g. a measurement of an accelerometer) which is not independent of another random variable X (e.g. the activity of the person). Thus, the utility of evidence y shall be determined by its impact on X. This impact is defined by:

$$U(X:y) = \underbrace{U(X|y)}_{\substack{\text{Utility of X} \\ \text{given known } y}} - \underbrace{U(X)}_{\substack{\text{Utility of X} \\ \text{not given } y}} \tag{5.68}$$

with $U(X) : X \to \mathbb{R}$ as the utility function which maps a random variable X to a real number. $U(X|y)$ is the utility function that maps the random variable X given evidence y to a real number. Since X is subject to uncertainty, equation 5.68 can be enhanced to calculate the *expected utility (EU)* gain by summing over all states weighted by their probability of occurrence:

$$\begin{aligned} EU(X:y) &= EU(X|y) && - EU(X) \\ &= \underbrace{\sum_{x \in X} U(x|y)P(x|y)}_{\substack{\text{Expected Utility of } X \\ \text{given known } y}} && - \underbrace{\sum_{x \in X} U(x)P(x)}_{\substack{\text{Expected Utility of } X \\ \text{not given any } y}} \end{aligned} \tag{5.69}$$

Unfortunately, it is not possible to determine y unless the Bayeslets are connected. The solution is to calculate the expected utility which can be gained from acquiring any $y \in Y$ instead. Therefore, one has to sum over the utility $U(x|y)$ of all possible outcomes of y, weighted not merely by $P(x|y)$ but also by $P(y)$, the prior probability of y:

$$EU(X:Y) = EU(X|Y) - EU(X) = \underbrace{\sum_{x \in X} \sum_{y \in Y} U(x|y)P(x|y)P(y)}_{\substack{\text{Expected Utility of } X \\ \text{given unknown } Y}} - \underbrace{\sum_{x \in X} U(x)P(x)}_{\substack{\text{Expected Utility of } X \\ \text{not given any } y}} \tag{5.70}$$

The equations above are defined in utility theory to rate the importance of a node Y for X with a known direction of their dependency. Bayeslets however represent sets of nodes calculating a posterior probability which is introduced as soft evidence to the input node of another Bayeslet. The substitute structure of *virtual evidence* for soft evidence allows the direct application of the concepts above however. As explained in section 2.3.2 (in particular Figure 2.8 on page 31), the soft evidence is mapped to the conditional probabilities of a virtual child node with hard evidence. The expected posterior distribution of Bayeslets is static and therefore maintained in the meta information of the Bayeslet specification. Hence we can always assume that the plugged-in Bayeslet is represented by a child node Y of an input node Y_X in the Bayeslet with the queried output node X.

To account for already acquired knowledge, e.g. from already connected Bayeslets, Eq. (5.70) can be enhanced to consider contextual knowledge c:

$$EU(X:Y|c) = \underbrace{\sum_{x \in X} \sum_{y \in Y} U(x|y,c)P(x|y,c)P(y,c)}_{\substack{\text{Expected Utility of } X \\ \text{given unknown } Y \text{ and known c}}} - \underbrace{\sum_{x \in X} U(x|c)P(x|c)}_{\substack{\text{Expected Utility of } X \\ \text{given known } c}} \tag{5.71}$$

$EU(X:Y|c)$ determines the utility which can be expected by the connection of a Bayeslet represented by Y given already acquired context c to input node X.

With Eq. (5.71) an intelligent decision maker could control the connection of Bayeslets as follows:

$$\begin{aligned} &\texttt{if } (EU(X:Y|c) > 0) \text{ use } Y \\ &\texttt{else retain } Y \end{aligned} \tag{5.72}$$

$EU(X : Y|c) \geq 0$ as will be shown in the following paragraphs. This means that additional evidence never decreases the utility and, thus, Bayeslets in most cases shall be connected. Therefore also the costs have to be taken into account which can arise from e.g. higher processing load, communication costs in case Bayeslets reside on remote entities or monetary costs if evidence from a remote Bayeslet is provided by a commercial service, e.g. a commercial weather service for highly accurate weather information.

To take into account costs that reduce the gross utility, Eq. (5.71) can be enhanced to the so called *Net Expected Utility (NetEU)*:

$$NetEU(X : Y|c) = EU(X : Y|c) - C(Y) \tag{5.73}$$

in case costs for all $y \in Y$ are equal. If this is not the case, the $NetEU(X : Y|c)$ is calculated by:

$$NetEU(X : Y|c) = \sum_{x \in X} \sum_{y \in Y} (U(x|y,c) - C(y))P(x|y,c)P(y,c) - \sum_{x \in X} U(x|c)P(x|c) \tag{5.74}$$

In the following, two different kinds of utility functions are proposed which can be used as criteria for the connection of Bayeslets similar to the concepts of information gathering and dissemination by Röckl in [102].

Probability-based Utility Functions

Often the utility of a random variable increases the "better it is known", i.e. the less the inherent uncertainty is. Also humans regularly acquire new information from independent sources if they are uncertain about the "true" state of an unknown process: If we are uncertain about the weather tomorrow, we check the recent weather forecast.

A suitable utility function would decrease (for instance logarithmic), if the added information decreases the certainty. The binary logarithm of the probability as used by Shannon in [105] fulfils these requirements. In this case $EU(X : Y)$ of Eq. (5.70) is equivalent to Shannon's Mutual Information $I(X : Y)$:

$$\begin{aligned} EU(X : Y) \equiv I(X : Y) &= \sum_{x \in X} \sum_{y \in Y} log_2 P(x|y)P(x|y)P(y) \quad - \sum_{x \in X} log_2 P(x)P(x) \\ &= -H(X|Y) \qquad\qquad\qquad\qquad + H(X) \quad \textit{[in bits]} \end{aligned} \tag{5.75}$$

with $H(X)$ being the entropy of the random variable X, $H(X|Y)$ being the conditional entropy of X given Y, see Eq. (5.59).

In this case, $EU(X : Y) \geq 0$ because:

$$\begin{aligned} EU(X : Y) &= -\sum_{x \in X} \sum_{y \in Y} P(x,y) \log \frac{P(x)}{P(x|y)} \\ &\geq^{(*)} -\log \sum_{x \in X} \sum_{y \in Y} P(x,y) \frac{P(x)}{P(x|y)} \\ &= -\log \sum_{x \in X} \sum_{y \in Y} P(x,y) \frac{P(x)P(y)}{P(x,y)} \\ &= -\log \sum_{x \in X} \sum_{y \in Y} P(x)P(y) = -\log 1 = 0 \end{aligned} \tag{5.76}$$

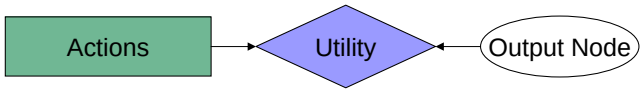


Figure 5.38: A *Decision Network* with one decision node, one utility node and one standard node representing a random variable.

As $(-\log)$ is a convex function [97], in the line in Eq. (5.76) marked $^{(*)}$, the Jensen inequality $E_x[f(x)] \geq f(E_x[x])$ for convex functions f [101] has been used.

To compute the net expected utility, a measure for the costs has to be specified. It can include memory consumption, communication, and CPU costs, depending on the needs of the implementing system.

Regarding CPU costs, i.e. inference time, the following could be an adequate measure:

$$C(Y) = H(Y) - \frac{H(Y)}{n_Y^a + b} \quad , \tag{5.77}$$

with the entropy of Y, $H(Y)$, $a \in \mathbb{R}$, $b \geq 1$ constants and n_Y the size of the biggest cluster in the junction tree [86] of the Bayeslet containing Y. $C(Y)$ has got the entropy of Y as an upper bound, to stay comparable with the mutual information. If the maximum cluster in the junction tree (which has large influence on the evaluation complexity, see [85]) is very small, the denominator of the second summand $n_Y^a + b \approx 1$ and the costs $C(Y) \approx 0$.

Decision-based Utility Functions

Although normally it is beneficial to head towards less uncertainty in the higher level information, in decision support systems utility shall be based on the outcome of actions instead of uncertainty reduction. Therefore the Bayeslet model can be extended by the notion of *action*, known from decision theory. Formally, Savage defines an action as *"a function attaching a consequence to each state of the world"* [104]:

$$\begin{aligned} a &: X \to X \\ a(x) &= x_a, \quad \text{with } x, x_a \in X \ , \end{aligned} \tag{5.78}$$

with x_a being the consequence of x after performing the action a.

To incorporate actions into Bayesian networks, the latter have been extended in decision theory to *Decision Networks* [66], also called influence diagrams [92, 62].

Decision networks introduce two new types of nodes, *decision nodes* and *utility* or *value nodes*. The values of each decision variable are the possible actions, they are imposed from the outside to meet some optimisation objective [62], utility nodes represent a utility or cost function. Within a decision network, decision variables are depicted as rectangular boxes and value functions as diamond-shaped boxes as shown in Figure 5.38.

A rational decision maker will choose the action which maximises the resulting utility. Thus, the expected utility can be replaced by the *maximum expected utility (MEU)* [66]:

$$EU(X|Y) \equiv MEU(X|Y) = \arg\max_{a \in A} EU(X|Y,a) = \arg\max_{a \in A} \sum_{x \in X} U(x|Y,a) P(x|Y) \tag{5.79}$$

If $EU(X|Y)$ is substituted by $MEU(X|Y)$ in Eq. (5.70), $EU(X:Y)$ is equivalent to the so called *Value of Information (VoI)* in information value theory [56]:

$$EU(X:Y) \equiv VoI(X:Y) = MEU(X|Y) - MEU(X) \tag{5.80}$$

Intuitively understandable, additional evidence never decreases the VoI. $VoI(X:Y) \geq 0$ can also be shown by the following proof:

$$\begin{aligned} MEU(X|Y) &= \sum_{y \in Y} P(y)\ MEU(X|y) \\ &= \sum_{y \in Y} P(y) \arg\max_{a \in A} \sum_{x \in X} P(x|y)U(x|y,a) \\ &\geq \arg\max_{a \in A} \sum_{x \in X} \sum_{y \in Y} P(y)P(x|y)U(x|a) \\ &= \arg\max_{a \in A} \sum_{x \in X} P(x)U(x|a) = MEU(X) \end{aligned}$$

Here, the conditional independence of the utility function $U(x|y,a) = U(x|a)$ and the Jensen inequality $E_x[f(x)] \geq f(E_x[x])$ [101] have been used.

5.4.4.3 Summary

Both approaches fit the requirements of dynamic composition very well. They use the expected evidence of a plugging candidate Bayeslet to measure the impact on either the output node itself or on actions relevant to the output node. The information necessary for this is static and therefore can be maintained in the Bayeslet specification and used without computation and communication overhead.

For the VoI, the design of the utility function and its harmonisation with the cost function for the composition is challenging and tedious, but adds flexibility to the system in comparison to pure uncertainty reduction. It would be the task of a human domain expert. A simplification is modelling the utility nodes as the reduction of uncertainty in the target random variable of the Bayeslet, as shown in Figure 5.39.

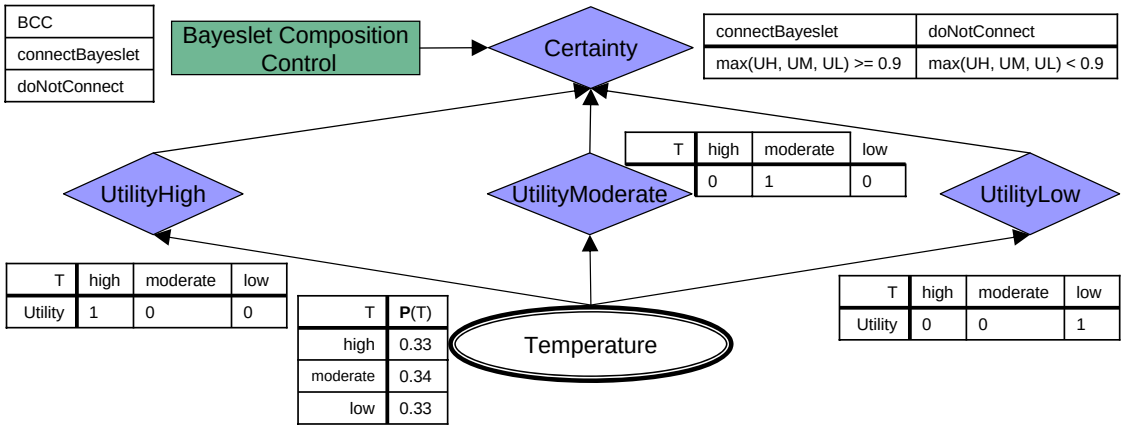


Figure 5.39: A decision network structure which can decide if any Bayeslet pluggable into the output node *Temperature* is to be connected. The utility is modelled to represent the expected certainty of the output node. The Bayeslet is connected if the expected certainty is above a certain threshold set to 90 % in this example.

In a two-layered structure of utility nodes, it can be decided if the certainty in the output node is above a fixed threshold. The lower layer has one basic utility node for each value of the output RV, representing the probability of this value. The upper layer checks if the maximum of the probabilities of each value is above a certain threshold. The actions *connectBayeslet* and *doNotConnect* are then taken based on this utility.

5.4.5 Context Inference with Bayeslets

The overall concept for Bayeslets offers approaches to reduce the number of nodes and the size of their probability tables regarding parents and value ranges. The performance gain achieved with these measures however depends also on the inference process used for evaluation.

This work has not focussed on finding a new inference algorithm for composed modules of Bayesian network, but shall select the most appropriate approach for the requirements of ubiquitous computing environments.

Therefore the next section summarises the requirements identified in the past sections, before section 5.4.5.2 repeats the available options, selects the most appropriate ones and presents adaptations as far as necessary. Bayeslets allow to evaluate an overall BN separately and hence even to distribute inference across different devices in the pervasive computing system.

5.4.5.1 Requirements for Bayeslet Inference

The inference approach selected for Bayeslets mainly has to fulfil the requirements (1) and (4) of section 3.3 along with those imposed by the last sections limiting the inference complexity. As such:

- inference has to be modular, so inference time is shorter.
- inference has to be performed locally to avoid remote communication of frequent evidence.
- inference must be able to incorporate the outcomes of different inference mechanisms.
- inference must not depend on the high update rates of low level context.
- inference has to cope with the dynamic modification of the value ranges suggested in section 5.4.3.
- inference needs to be flexible in using prior probabilities, stored evidence or Bayeslet inference for input nodes.

Modular probabilistic inference will always decrease inference time due to the "divide and conquer" principle – in terms of the PPTC approach presented above, the construction of smaller junction trees, smaller CPTs and less nodes to be processed decreases inference time.

In addition, modularity is useful, if Bayeslets can be evaluated separately on different machines. Not only are different processors loaded in parallel, in addition this allows for access control and encapsulation of private information.

Requirement (4) from section 3.3 demands "support for different characteristics of context", in particular with regards to change frequency, but also concerning the integration of context information from non-Bayesian inference modules.

In section 2.2, probabilistic context inference was shown to satisfy the requirements best. However it is easily understood that for different inference goals and different input data, different inference approaches may be most suitable. Thereby it is the target that all types of inferred context can be used – not only by end consumers of the inferred information, but also by other inference algorithms. Therefore the necessary context meta information has to be provided, in particular the uncertainty of an inference outcome.

All probabilistic algorithms, static or dynamic, will inherently provide a posterior probability distribution as uncertainty measure. Non-probabilistic inference algorithms also have to provide a probability distribution over the inferred states in the context meta information. If they did not, hard evidence would have to be assumed for the inferred value, which however would undermine the advantages of reasoning under uncertainty and might heavily falsify the results.

With regards to the change frequency of different context aspects, section 3.2 has given insight. Change frequencies may range from hundreds of Hertz until only few per hour or even less. This has direct impact on the choice of the most appropriate inference algorithm and is closely related to the level of context information. Figure 5.40 shows an information pyramid, distinguishing information by their frequency.

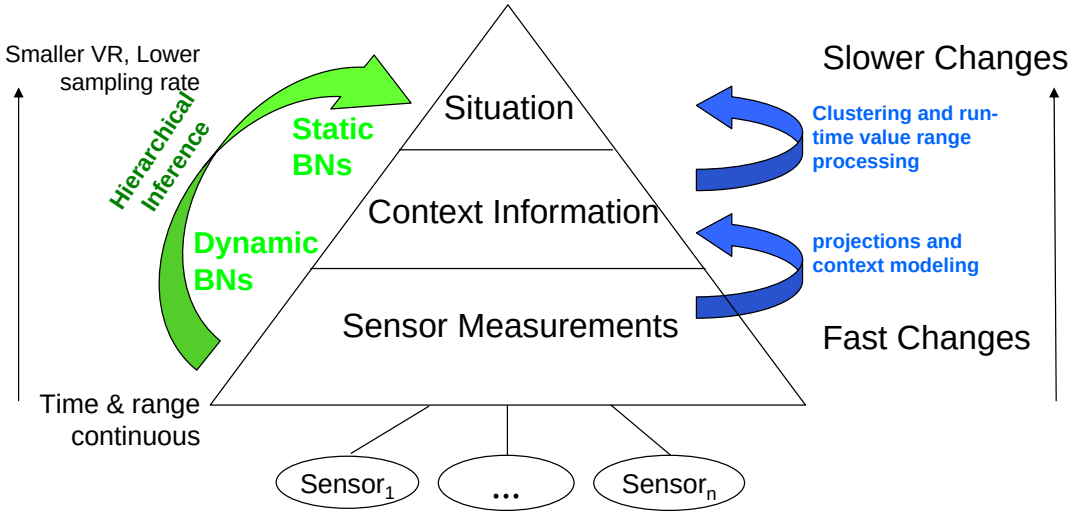


Figure 5.40: Hierarchical context inference takes into account the different qualities of input data. Raw data at high frequency and high-level information require different processing, but these inference methods have to be compatible.

At the bottom of the pyramid in Figure 5.40, we have a large spectrum of different raw data, which can come with frequencies up to several 100 Hz (as the IMU for activity recognition). This magnitude is very high compared to the human step (frequencies around 1 Hz) which serves as our reference system. High frequent changes are also less meaningful in many cases. For instance, acceleration data of a user's foot would not usually be of direct interest for a ubiquitous computing framework. Based on those information however, a step can be calculated, which changes only with a frequency in the order of 1 Hz. A step can already be of relevance for positioning systems or movement detectors (as could be seen in section 5.2.2), but only the subsequent processing of the data, one layer above in the pyramid, would be of direct use to all users of the ubiquitous computing framework: from the step, one can calculate the current position based on an absolute starting position.

Furthermore computation of consequences and events with so high frequency is hardly feasible in a computing framework, even storage and management would impose a large burden on any CMS. Also the value range is often very fine grained or continuous where tiny value changes do not have impact on higher level context, but complicate computation based on it. Reconsidering the definition of context information presented in Def. 1 on page 15, such raw information cannot be used to characterise the state of an entity – and is hence not yet context information. However, it can be "refined" to context information by context inference. Often, the most suitable inference algorithms for such low level

information incorporates the dynamics of the to be inferred context information, making use of the high update rates.

It can be observed that higher level context, like activity, the season or a mood, change less frequently and can be described with less different states which differ then significantly in their semantics. Where changes are less frequent, static inference approaches are often equally well suited, but more efficient.

The combination of dynamic and static Bayesian inference methods like those presented in 2.3.3 and 2.3.6 are suitable candidates for inference on all levels of information. They cope with different requirements of different data types, and share the same requirements for input information and produce equivalent output with a posterior probability distribution.

Context Inference with Bayeslets has to consider these observations. It has to dissolve the computation of higher level context from the computation of lower level context by separate evaluation of different Bayeslets. Like that, inference rule modules can be evaluated with the most suitable inference algorithm regarding the characteristics of the information contained in it. Compatibility with other modules is ensured by specifying the optional input from other modules and provision of a posterior distribution over the possible states of the target context type. Typically, dynamic Bayesian networks will be used to model the inference of lower level context, static Bayesian networks for high level context, where the DBN is largely independent from the high level context while the static BN uses the posterior of the DBN as evidence.

To cope with the dynamic value ranges, inference has to maintain the filtered probabilities $\mathbf{P}_f(V_i)_t$ of the output nodes V_i at all times t and trigger repartitioning of the network when necessary.

Just like the value ranges are partitioned before inference starts, also the Bayeslets relevant for an inference target are selected and their combination is clear. Important here is however, that inference can treat the Bayeslet's input nodes always in the same way, no matter if there is already recent evidence for the input node, if a Bayeslet for inferring the node can be connected, or if the prior (conditional) probability of the input node is used for inference.

5.4.5.2 Inference Approach in modular Bayesian networks

Literature has proposed many different approaches for probabilistic inference with structured Bayesian networks.

Many research groups investigate the usage of exact inference processes like [265], [90] or [94] in Bayesian network. To be able to do so, they have to be able to make assumptions about the dependencies of the substructures, hence limit the possible connections of inference modules. The substructures thereby are completely separated and mutually have to wait for the outcomes of other modules.

Other approaches with exact inference do not assume complete separation of the modules, just like Bayeslets where different Bayeslets overlap with their input and output nodes. An example for these approaches are MSBN [273] or [87]. For inference, still common hyper structures are created for inference of the joint network.

A different approach, approximating only the actual inference outcome uses soft evidence to connect inference modules, e.g. [253]. The evaluation outcome of a BN module is introduced as soft evidence in specified nodes of a requesting BN module. The requesting module thereby has to wait with its probability propagation for the output of the linked module. This approach neglects the exact computation of the posterior probability and

makes "backward" inference (influence from the requested to the linked BN module) impossible.

Restrictions on the connection types between Bayeslets and the mutual exclusive separation of Bayeslets are not feasible in ubiquitous computing, where different parties provide Bayeslets as inference rules and different Bayeslets (e.g. sensors) provide input to the same random variable.

Also inference with a common hyper structure is not appropriate as this impedes distributed local inference without further connections between Bayeslets.

The only viable approach is constituted by *soft inference*. This approach implies some loss of inference precision and the proposed evaluation is only approximate inference. Soft evidence, introduced as result of one Bayeslet into the input node of another one, does not d-separate both Bayeslets, neither when the input node is ancestor nor sibling of the queried output node.

The selection of an approximate inference approach is appropriate. By the structural modifications of the Bayeslet proposed for the separation of Bayeslets, reduction of parents and modifications of value ranges, every inference result is an approximation of the posterior probability. With the requirements stated above hence, Bayeslet inference has to be in the class of *model simplification* inference algorithms (see section 2.3.3.3).

The performance gain with this approach does not only stem from the possible distribution on different devices, but also from the fact that smaller networks are inferred upon – the *divide and conquer* principle. Moreover it is the only approach allowing for the inclusion of other inference algorithms.

Assuming static Bayesian networks, some measures can be taken to further optimise inference in every single Bayeslet:

- All nodes which are not in the Markov boundary of the requested output node are pruned.
- The network structure is analysed and the most appropriate inference approach is selected. I.e. the inference algorithm with the lowest complexity is searched, e.g. in tree structured network structures complex algorithms like PPTC do not have to be applied.
- For multiply connected BN structures, the junction trees with their cluster probability tables can be computed once right after network creation and used then for every posterior estimation, as they are static.

5.4.6 Discussion

Section 5.4 has described the proposed concept for the application of modular Bayesian inference to context inference in ubiquitous computing. From the literature it has adopted the idea of structuring large Bayesian networks to allow for efficient inference in them. The proposed adaptation to context awareness realises also the personalisation and heterogeneity required for ubiquitous computing by means of Bayeslets.

The most effective way to reduce the number of parents in a BN is personalisation. Not the same dependencies hold for all persons – so only a personalised Bayeslet can use only the relevant dependencies. Everything that is not relevant for the user is neglected.

This reduces the size of the CPTs for inference, together with the proposed context aware reduction of value ranges. The proposed novel approach adopts different mecha-

nisms for discretisation of continuous random variables and transforms them to fit context inference in discrete Bayesian networks.

The approach presented for the composition of Bayeslet is very innovative. It concretises the rather vague concept of *relevancy* mentioned in most definitions of context and allows the inference of actual context, whereas monolithic Bayesian networks would only infer the general situation (cf. Definition 5 on page 16). For such composition of Bayeslets, approaches from information and decision theory have been adapted to the Bayeslet concept. Both approaches use slightly different points of view and can be used both in the respective situations.

To infer high level context, the combination of different inference methods is proposed here, each one fitting best the requirements of the represented model, in particular also a combination of static and dynamic Bayesian methods.

The approach allowing best the integration of all requirements is to separately infer each Bayeslet and to introduce the computed posterior probability as soft evidence in the connected Bayeslet. This is an approximate inference approach belonging to the class of *model simplifications*. A general shortcoming of this approach is the limitation of inference in one direction, i.e. allowing deductive, but not inductive inference. The restriction of Bayeslets to specify input and output nodes however already anticipates this reduction, in order to allow for more effective and cycle free composition.

All in all, the Bayeslet concept imposes some limitations to monolithic Bayesian networks and its inference:

- Restricting the interface of a Bayeslet to only predefined input and output nodes is reducing the flexibility of the use of Bayesian networks.
- Therefore the inference in a combined Bayeslet network is suffering an (if only moderate) precision loss, as one can consider it as missing some edges, in comparison to the monolithic approach with all possible dependencies modelled.
- The necessary assembly step takes time and adds complexity to the context management system.
- Also personalisation of the random variables, in particular the value range reduction, adds further complexity to the CMS which has to permanently monitor conditions and trigger the respective actions.

These limitations however are outweighed by the following advantages:

- Bayeslets represent enclosed and encapsulated knowledge domains that are, due to their size, easily manageable and processable.
- Bayeslets allow for parallel evaluation and partial reevaluation.
- Bayeslets allow for personalisation of inference rules.
- Bayeslets give control of information access to the user.
- Bayeslets allow for efficient integration of information coming from different domains.
- Bayeslets can be learned automatically or be provided by external application developers. This enables the latter to adapt the system's capabilities to their needs on the fly.

The above limitations have to be accepted to allow the stronger advantages which allow for tractable, adaptive context inference in ubiquitous computing environments.

Chapter 6

Application and Evaluation

This chapter demonstrates how the different concepts presented in chapter 5 are applied to context inference in ubiquitous computing and evaluates their usefulness in example scenarios. To this end, the structure follows the outline of chapter 5 as far as possible, emphasising the main contributions of this thesis.

Section 6.1 refers to the concepts of section 5.2. It reports on the results of the efficient fusion of Wi-Fi fingerprinting and a foot-mounted INS, as well as on the conversion of these results to symbolic location used in high level context inference.

Section 6.2 presents the performance results of the different approaches for the recognition of human motion related activities introduced in section 5.3. The naïve Bayes model and the learnt structure model, as well as static and dynamic inference methods are compared.

Sections 6.3 and 6.4 then give application examples for the main innovations of section 5.4, dynamic value ranges for random variables in Bayeslets and situation specific determination of relevancy for the composition of Bayeslets.

Finally, section 6.5 demonstrates the overall added value of the concepts from chapter 5. Context inference in an example scenario with current state of the art techniques is compared to the approach proposed in this thesis with regards to computation and communication demands.

6.1 Absolute and Symbolic Positioning

This section evaluates the approaches to determine absolute and symbolic location presented in section 5.2, pp. 87 et seqq.

6.1.1 Evaluation of Inertial and Wi-Fi Fingerprinting Location Fusion

In section 5.2.2, a process for the fusion of Wi-Fi fingerprinting and inertial measurements with an extended Kalman filter for efficient location estimation has been proposed. It was evaluated in the project IST Daidalos 2 in a building of the university of Aveiro, Portugal. The results have been described by the author et al. in [151].

6.1.1.1 Test Environment

To test the fusion methodology presented in section 5.2.2, a building of the university of Aveiro was chosen. It is equipped with eleven Wi-Fi access points on one floor as can be seen (black crosses) in Figure 6.1 for public internet access, as well as for research purposes in the labs. Detection of different offices and rooms was expected to be fairly easy by Wi-Fi

fingerprinting as infrastructure (in particular walls) should produce a significantly different fingerprint than in other rooms. Testing therefore concentrated on the circular corridor.

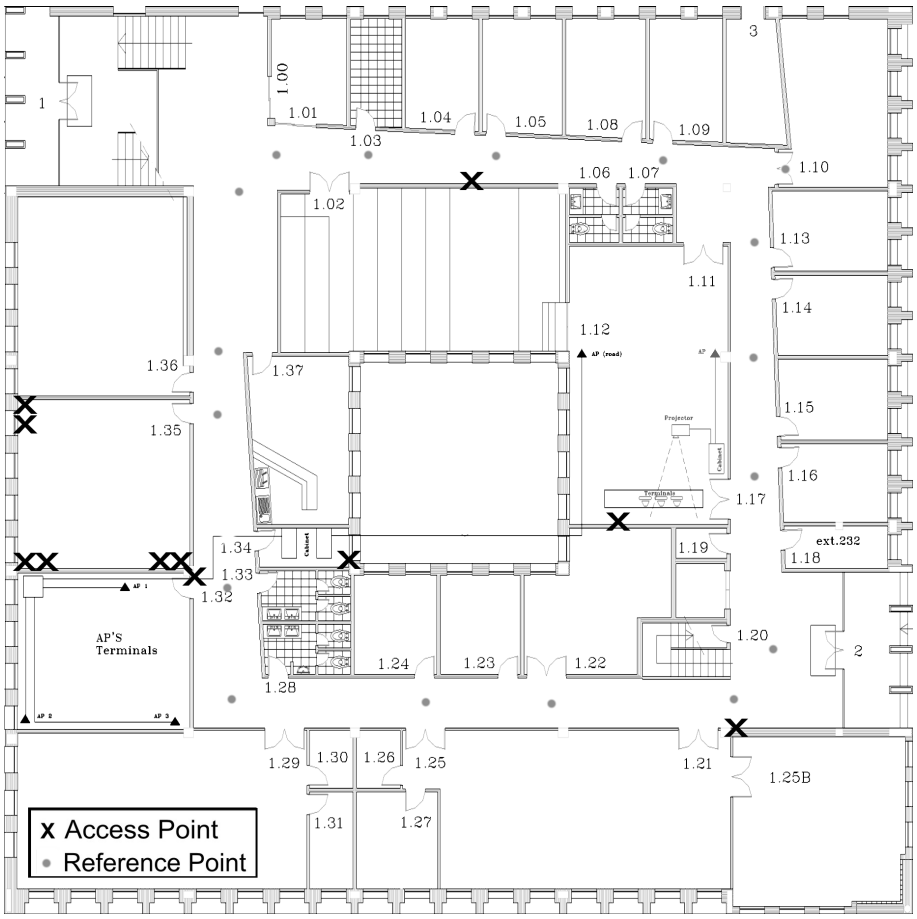


Figure 6.1: Floor plan of the test building with eleven access points (black crosses) and 17 reference points (grey dots).

As ground truth, 17 reference points in the corridor have been defined and marked on the floor of the building. Their location was measured in the building and transferred to metres on a floor plan with a known scale. The test users followed precisely the path defined by the reference points on a circular walk (four laps, taking about 280 s). Every time a reference point was passed, a button was pressed to record the timestamp.

Fingerprinting calibration points were taken approximately every 2.5 m in the corridor, with signal strength samples being taken once per second. Measurements were taken by holding a laptop at a fixed height (approx. 1.2 m), with slight motion to build up a PDF over a small region. A sample size of 60 was used for each calibration point. At each point, a minimum of three access points are always recorded. An example of the output for a single calibration point for three particular access points can be seen in Figure 6.2. The calibration point was recorded on the electronic floor plan to designate the physical location.

The implementation of the positioning system was realised in three separate subsystems, the fingerprinting, the step estimation and the fusion system, which were installed on two different laptops. They communicated in client-server mode as both sensor connections

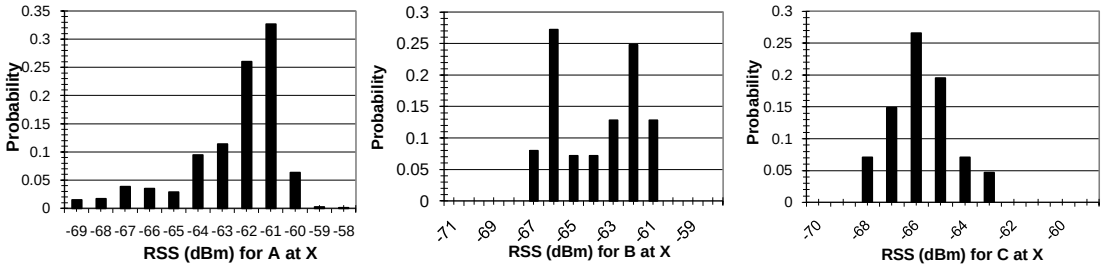


Figure 6.2: Example PDFs of one calibration point X for different access points A, B and C.

were implemented in C while the higher level extended Kalman filter was implemented in Java, as well as a visualisation application.

Laptop 1, with a Windows OS (for driver reasons), implemented the INS connection, the low level filter and the client part of the connection. Laptop 2, using Linux, hosted the fingerprinting subsystem, the server part of the connection, the high level EKF and the visualisation application. It was equipped with two wireless network cards for fingerprinting. One network card was used for scanning to ensure consistent results during the training stage, and to allow channel hopping without disrupting network communications. The second wireless card was used to send real-time location updates to other displays for live presentation of data. To reduce network related delays, both laptops were connected with an ethernet cable throughout the track.

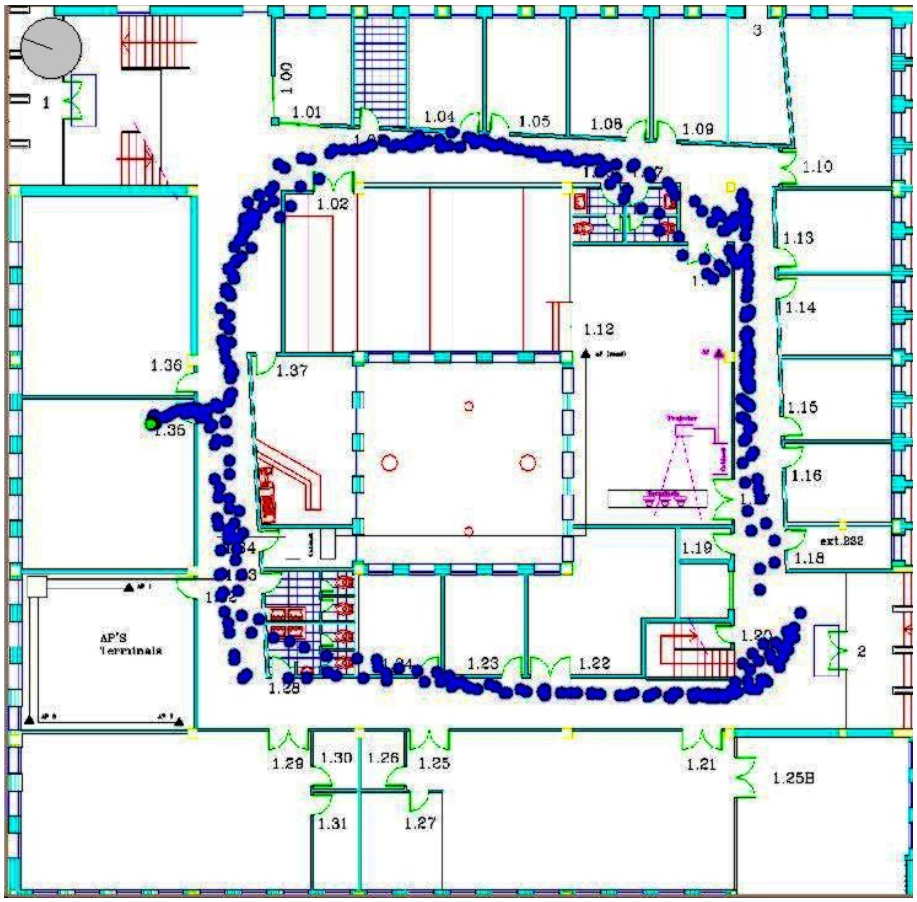


Figure 6.3: Visualisation of the recorded track.

The fused absolute position is passed from the Kalman filter to the visualisation application. Its output can be seen in Figure 6.3. Raw INS output on Laptop 1, as well as input (from fingerprinting and step estimator) and output of the sensor fusion on Laptop 2 have been logged with their timestamp in milliseconds. Together with the timestamps logged while passing a reference point marked on the floor, these logs have been used for the evaluation presented in the following subsection.

6.1.1.2 Accuracy of Positioning

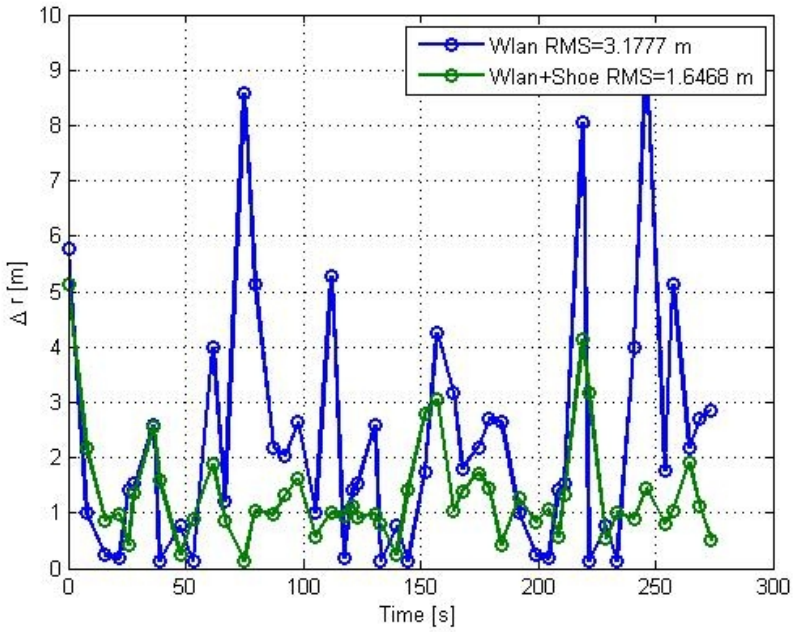


Figure 6.4: Position error (in metres) distribution over the whole walking time.

With the efficient EKF, different location sources can be fused to provide more precise absolute location. This section compares the accuracy of positioning with Wi-Fi fingerprinting alone and fused with the INS system in the environment presented above. Using INS alone would lead to significant drift over time with increasing errors. Therefore it is not a sensible standalone approach and not evaluated specifically.

To evaluate the logged data, the pedestrian's position calculated with Wi-Fi and the fused position were compared with the known, absolute position of the reference points at the moment when it was visited. The measure for the accuracy was the Euclidean distance (line of sight) of the calculated positions to the reference point in metres. Figure 6.4 shows those results for each of the 49 recorded points. Results for Wi-Fi fingerprinting standalone are presented by the blue line, the fused position error is shown with the green line. It is obvious that the fused result is influenced strongly by the fingerprinting result, in particular with regards to the fingerprinting errors, when wrong sample points were identified as current position and hence the error was particularly high. From both techniques we calculated the arithmetic mean error for our test track:

- 3.18 m for standalone Wi-Fi fingerprinting,
- 1.65 m for the fusion of fingerprinting and INS.

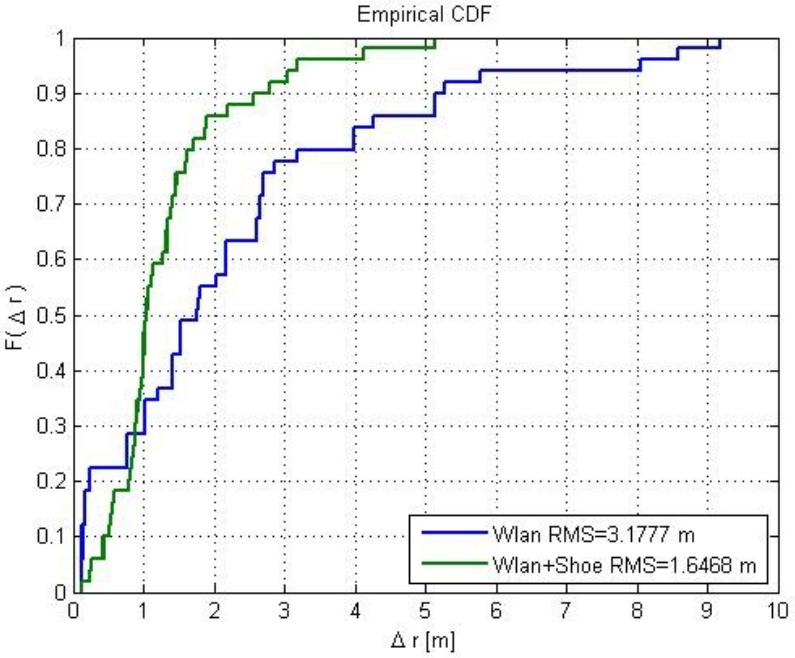


Figure 6.5: Cumulative probability distribution F of the position error in metres.

It can be seen that the fused result is more stable than the fingerprinting approach alone: the INS is preventing big jumps that are possible with pure fingerprinting. Standalone fingerprinting provides very good results when the reference points were very close to correctly detected fingerprinting sample points. This is visible in Figure 6.5 where the cumulative probability distribution is shown: fingerprinting provides more than 20 % of errors below half a meter, but also 20 % of errors over four metres – which appear in the fused results only with a frequency below 5 %. It can be seen that the majority of the calculated errors of the fusion lie around one meter, i.e. about the length of a step.

6.1.2 Time Complexity of Symbolic Location Determination

Section 5.2.3 on page 94 proposes a way to extend the estimation of absolute location to estimate also the symbolic location. This process is verified for a particle filter based approach which already incorporates maps and does not add any requirements hence.

6.1.2.1 Evaluation Environment

The proposed approach has been verified in an environment for distributed indoor/outdoor positioning with an existing data set which has already been used for evaluation of different location estimation techniques, for instance in [164].

The environment can flexibly plug-in different types of sensors and information and use different sequential Bayesian estimation techniques and movement models. For this evaluation, both indoors and outdoors GPS signals have been used together with the inertial measurements of an INS, measurements of an electronic compass and information about the location of the 57 walls defining the ground floor of an office building in a likelihood particle filter without movement model (see [164] for details).

A test user was equipped with the mentioned sensors and requested to walk along a specified path, visible in Figure 6.6, passing through predetermined ground truth points.

The ground truth of the walk recorded in the data set was measured with sub-centimetre accuracy using a tachymeter, a Leica Smart Station (TPS 1200), employing optical distance and angular measurements based on differential GPS for initial positioning.

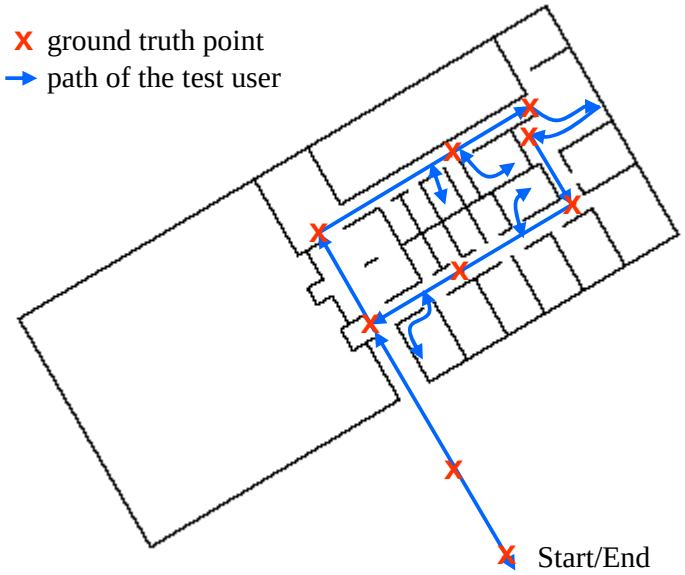


Figure 6.6: Path of a test user in the evaluation data set used to demonstrate the extraction of soft symbolic location during the location estimation process for highly precise absolute position.

The data recorded during this walk cover 588 second during which the test user enters a building after having passed two ground truth points outdoors. In the building, he walks three laps along the corridor of the ground floor of the office building entering the same five offices on every lap. After the third lap, the test user leaves the building.

Whenever the user passed through one of the ground truth points, the estimated position was compared to the true position. Errors between the true positions and the estimated pedestrian positions have been evaluated based on the fusion of the selected information in a particle filter with 2000 particles. In [164], it has been shown that the position error in this approach amounts to 1.5 m on average.

To evaluate the estimation of symbolic location together with highly precise absolute location in this environment, 23 symbolic locations have been defined as polygons in an XML format. The symbolic locations describe the main areas inside and outside the building (rooms, corridor, combinations of areas) which are all non overlapping in this example.

The evaluation was run on a computer with Intel Core2 Quad Q6600 processor at 2.4 GHz with 2 GB RAM, employing the Java implementation of the test environment under Windows XP Professional SP3. Walls and symbolic locations are thereby separated into different R-Trees in order to accommodate for their different treatment.

6.1.2.2 Performance of Symbolic Location Estimation

As described in section 5.2.3.1 on page 94, an R-tree is used in the implementation to constrain the number of currently relevant spatial objects. The 23 symbolic locations are therefore inserted into the R-Tree. Averaged on 50 repetitions, the construction of the R-Tree takes 2.05 ms ($\sigma = 4.08$). As the construction is a singular event preceding the location estimation, it does not influence the overall estimation time.

The R-Tree is used during the evaluation to determine the location of every particle. Instead of checking every symbolic location whether it contains the particle, the following approach can be taken:

1. Query the R-tree with the absolute position of the particle.
2. Check only the resulting symbolic locations.

The second step is necessary, as the R-Tree does not contain the polygons of the symbolic location themselves, but their bounding boxes. As a consequence, the bounding boxes of the rooms in Figure 6.6 overlap. In the example walk, on average $n = 2.67$ ($\sigma = 1.19$) symbolic locations are returned for the coordinates of a particle by the R-tree in about 1.56 million requests. The request itself thereby takes $r = 3.353$ µs ($\sigma = 8.283$).

Checking if a coordinate is contained in a polygon using the Java standard function (`javax.swing.JComponent#contains`) takes $c = 0.691$ µs ($\sigma = 6.286$), averaged over about 4.17 million method calls. The additional time of the R-tree request hence is justified if it is smaller than the time saved by the reduced number of relevant symbolic locations. In the present example, the overall number of symbolic locations has to exceed 8, assuming the measured average times for c, n, r.

During the 588 s of the walk, the system makes 782 estimations of the absolute and symbolic location, each taking on average 0.635 s for 2000 particles. The steps added to produce a soft estimation of the symbolic location take on average 4.23 µs ($\sigma = 15.03$) per particle. Summing the time of the 2000 particles, the time spent per location estimation on the calculation of symbolic location takes 8.47 ms and hence only 1.33 % of the overall estimation time.

6.1.3 Discussion

In both scenarios, the evaluation has shown that there are tractable location inference techniques providing highly accurate results for absolute location and calculating the symbolic location necessary for high level context inference without significant increase in resource consumption.

The Kalman filter based location fusion approach for a foot-mounted INS has been demonstrated to yield accurate results without high computational complexity in combination with a further absolute positioning method.

Without a further positioning method, INS and information from a map yield accurate results as shown in [164]. The results above show that the same map can be used to efficiently extract symbolic location information.

In a reasonably small environment like an office building the time complexity hardly exceeds 1 % of the overall estimation time. Such dimensions of floor plans are not unrealistic given the concepts of smart spaces. Not all worldwide available information are managed in a single system, but only locally relevant information are offered by smart spaces which makes the amounts of data easier to manage.

However also larger amounts of symbolic location information can be managed in such systems. Maps, e.g. such from OpenStreetMap[1], can be managed in efficient geospatial databases, as for example shown in the Osmosis[2] extension. The search complexity increases with higher numbers of symbolic locations, but in the ideal case (no overlapping bounding boxes for the queried position) only logarithmic to the base m, the minimum number of entries per node [158].

[1] `http://www.openstreetmap.org/`
[2] `http://wiki.openstreetmap.org/wiki/Osmosis`

6.2 Human Motion Related Activity Recognition

This section evaluates the approach to recognise the human motion related activities *sitting, standing, walking, running, jumping, falling* and *lying* according to the works of the author et al. in [211, 237, 209, 210]. The four different classification methods presented in section 5.3, pp. 107 et seqq., static and dynamic Bayesian inference based on a learnt BN structure and Naïve Bayes structure, are compared in subsection 6.2.2 based on the data set presented in section 5.3.3.1. Subsequently, in subsection 6.2.3 the computational complexity of the presented system is evaluated.

6.2.1 Evaluation Settings

Base for the evaluation are the data set recorded with the xsens MTx IMU described in section 5.3.3.1 and the Java implementation of the activity recognition system with the four alternative estimators and a visualisation module shown in Figure 6.7.

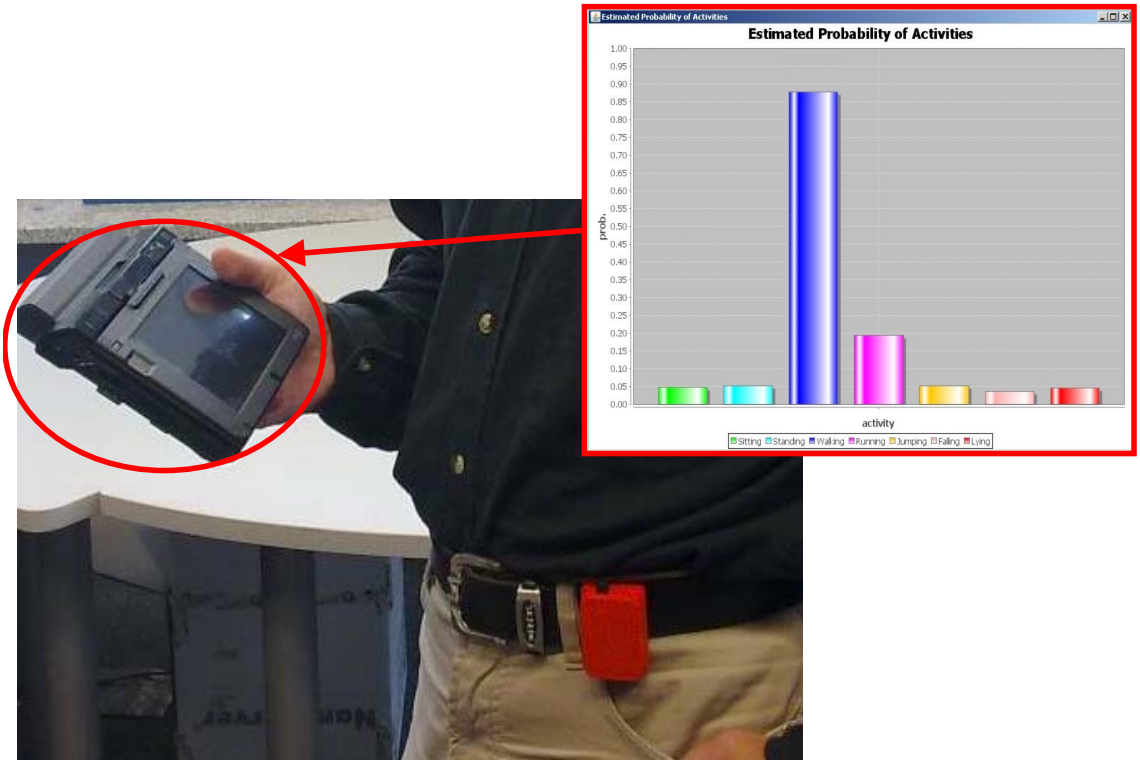


Figure 6.7: Implementation of the activity recognition system. The orange xsens MTx IMU is worn at the belt and connected to a mobile computer (in the red ellipse) which estimates the human motion related activity and and visualises the result with the probability bar chart shown in the top right corner.

The implementation follows the information flow explained in Figure 6.8. The acceleration and turn rate data in 3D and the rotation matrix are the input to the recognition algorithm which computes the basic signals. Using those signals, the features are calculated. These (and the signals which are used directly as features without any further computation, see section 5.3.2) are input to the static classifiers estimating the current activity. In the case of dynamic inference, this estimate is used in the dynamic classifier.

The results have been evaluated with a four-fold cross validation. In this evaluation approach, the data set is split into four parts of which always three are used to train the classifier and the fourth is used to test the trained classifier. This technique avoids the overlap of test and training sets. The average of the four tests gives the result.

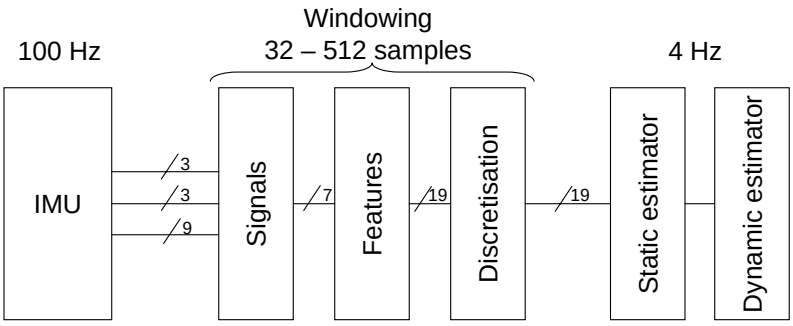


Figure 6.8: Information flow of the raw data to the finally recognised activity.

The results are measured in *precision* and *recall* rates, see [222]. The recall of class i, $Recall_i$, measures the rate of correct evaluations of a class i among all samples of that class:

$$Recall_i = \frac{TP_i}{TP_i + FN_i}, \tag{6.1}$$

while the precision of class i, $Precision_i$, measures the rate of correct evaluations of class i among all evaluations to that class:

$$Precision_i = \frac{TP_i}{TP_i + FP_i}. \tag{6.2}$$

True Positives (TP_i, elements of class i recognised as class i), True Negatives (TN_i, not elements of class i not recognised as class i), False Positives (FP_i, not elements of class i recognised as class i) and False Negatives (FN_i, elements of class i not recognised as class i) are used to compute these measures.

The first part of the evaluation compares the recognition results of two persons, Emil and Sinja, with the labelled ground truth. The raw data of the IMU had been recorded, but not used for training the classifier before the recognition algorithms had been applied. The data rate of the recorded data is 100 Hz, the inference rate 4 Hz.

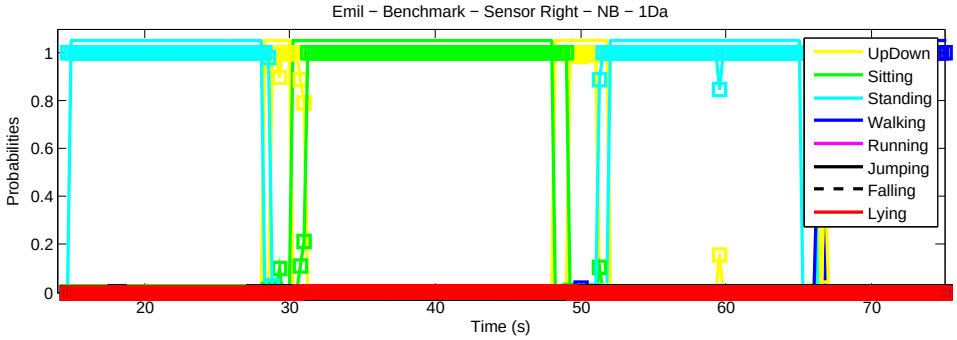
An error source, appearing in this comparison as well as in precision and recall, originates from the manual labelling of the test data which appears in particular at the transitions between activities.

The system's accuracy for transitions (going up and down) is not evaluated in this section. They have not been a recognition target for the system, but only used as additional states to improve the dynamic inference.

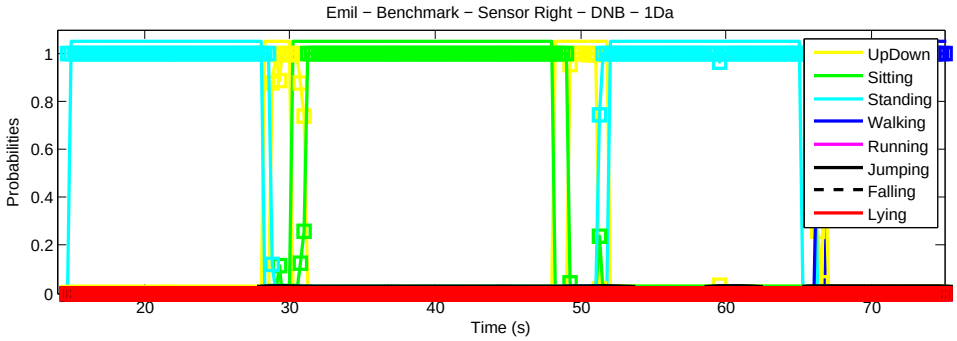
The last part of the evaluation measures the execution time. To this end, feature computation and inference were repeated 780 times. The evaluation platform was a PC with Intel Core 2 Duo E8400 processor at 3.0 GHz with 2 GB RAM running Windows XP.

6.2.2 Inference Results

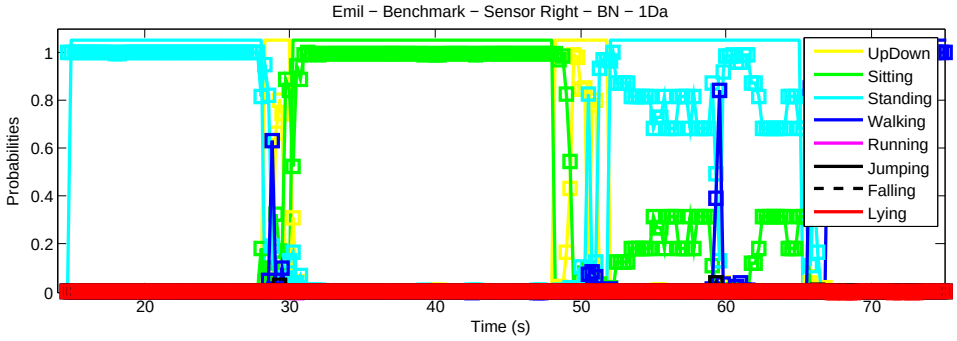
This section presents the results of the inference system and compares the four approaches for inference. First, sequences of activity recognitions are compared to the ground truth to analyse difficulties of the estimators and to compare their behaviour in Figure 6.9, 6.10 and 6.11. Then the results of the recognition system in a four-fold cross validation are given in terms of precision and recall in Figure 6.12. As a last step, the effect of a recognition delay on the precision and recall values is analysed in Table 6.1.



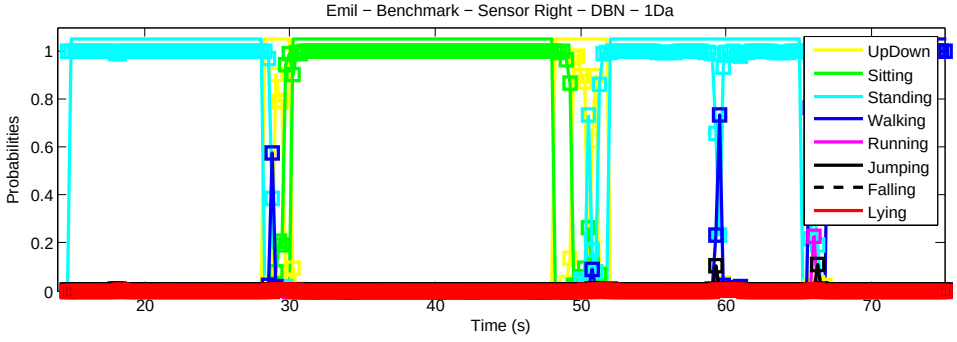
(a) Static estimator based on Naïve Bayes structure



(b) Dynamic estimator based on Naïve Bayes structure

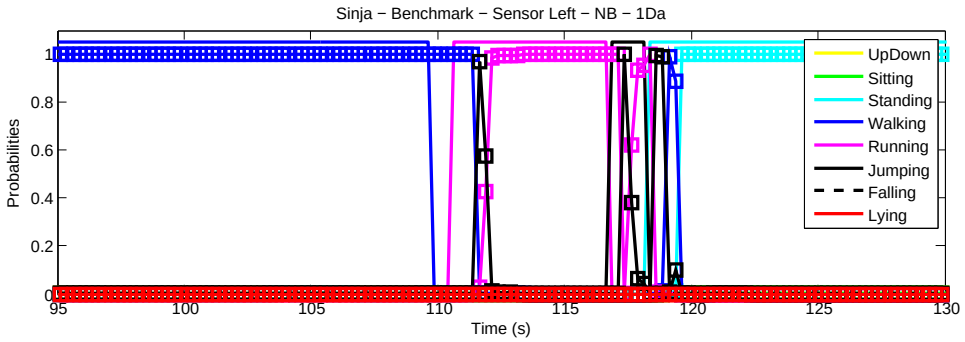


(c) Estimator estimator based on learnt BN structure

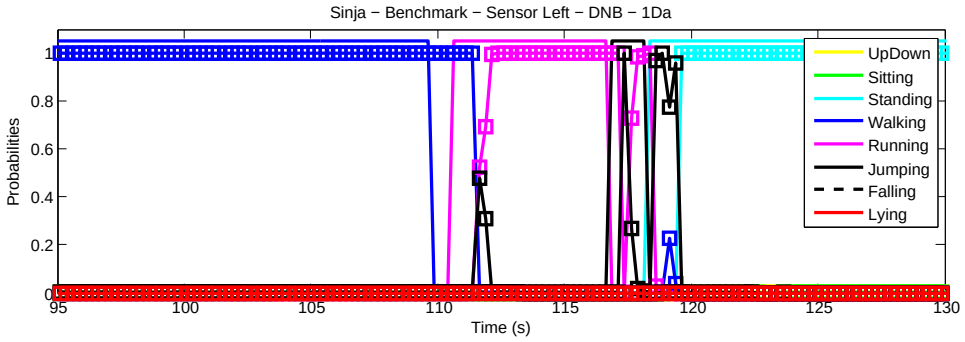


(d) Dynamic estimator based on learnt BN structure

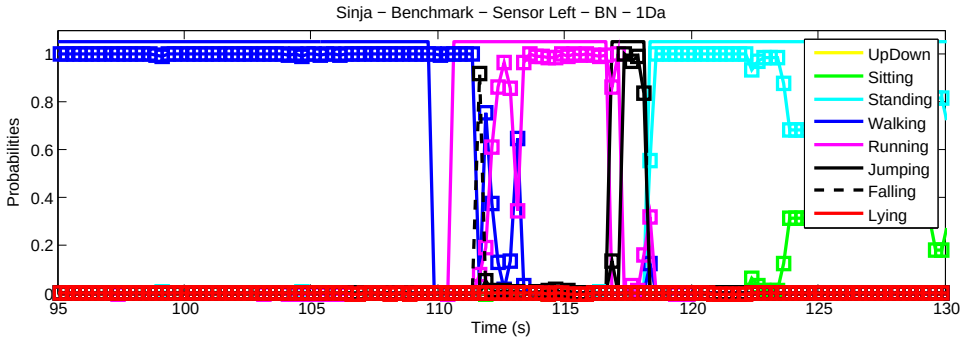
Figure 6.9: Inference results for an example of the sequence *standing, sitting* and *standing*. The thin, coloured line at the top of these figures depicts the ground truth, colours identify the current activity according to the legend on the right. Below the ground truth, the estimated probabilities of every activity are indicated with the coloured squares [209].



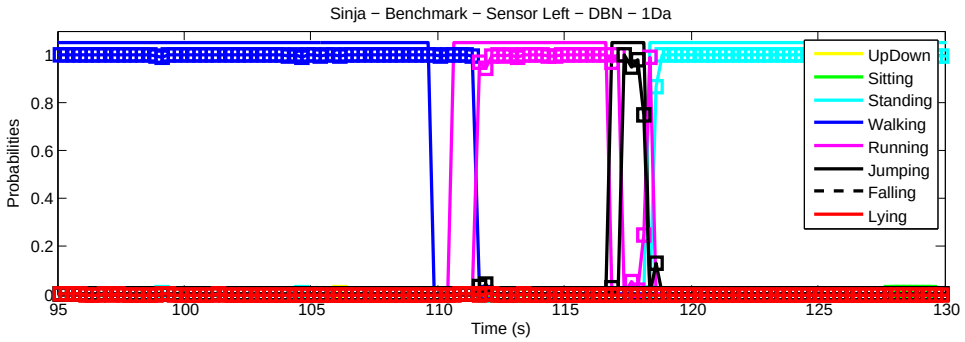
(a) Static estimator based on Naïve Bayes structure



(b) Dynamic estimator based on Naïve Bayes structure

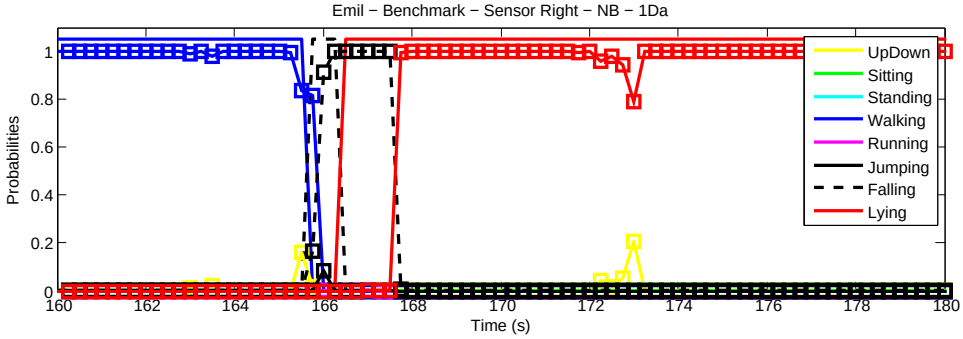


(c) Static estimator based on learnt BN structure

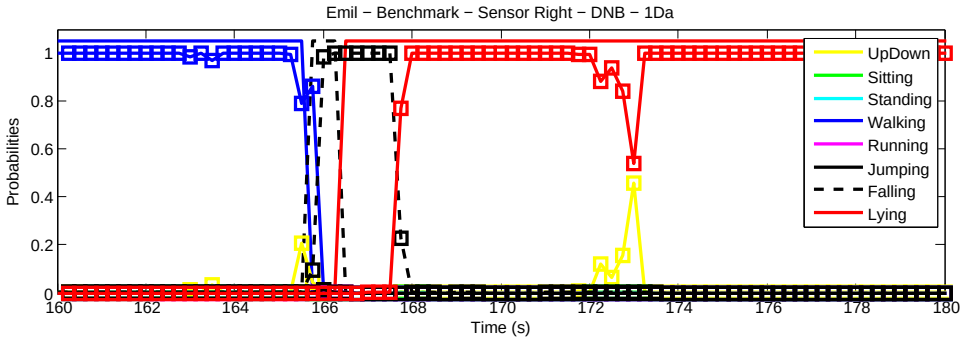


(d) Dynamic estimator based on learnt BN structure

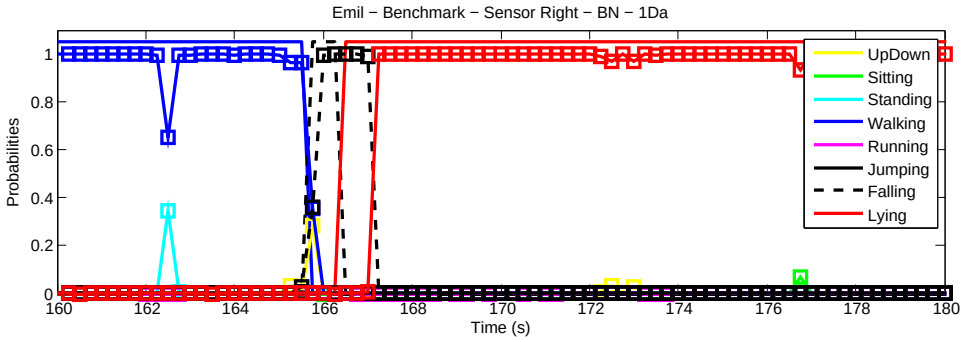
Figure 6.10: Inference results of the sequence *walking, running, jumping, standing* [209]. The thin, coloured line at the top of the figures depicts the ground truth, colours identify the current activity. Below the ground truth, the estimated probabilities of every activity are indicated with coloured squares.



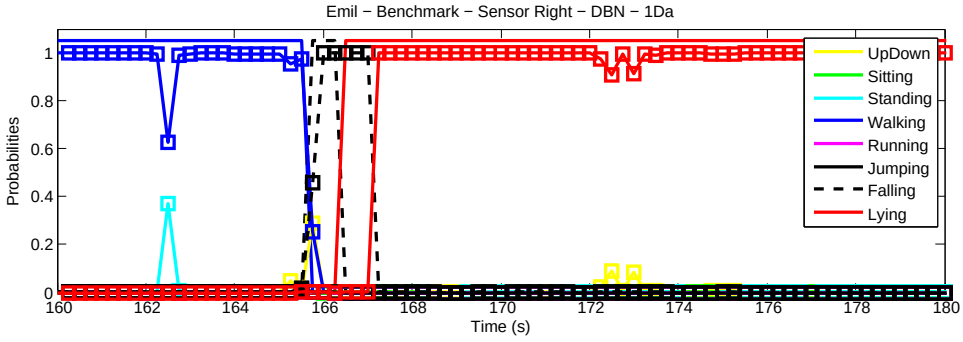
(a) Static estimator based on Naïve Bayes structure



(b) Dynamic estimator based on Naïve Bayes structure



(c) Static estimator based on learnt BN structure



(d) Dynamic estimator based on learnt BN structure

Figure 6.11: Inference results of the sequence *walking, falling, lying* [209]. The thin, coloured line at the top of the figures depicts the ground truth, colours identify the current activity. Below the ground truth, the estimated probabilities of every activity are indicated with coloured squares.

The evaluation of the activities *standing, sitting* and *standing* of Emil is shown in Figure 6.9. The distinction between *standing* and *sitting* is based only on the attitude of the sensor which depends again on the particular way the subject is sitting and is therefore difficult to detect with the available features, compare section 5.3.2.3. It can be seen however that the dynamic estimation with the transition model described in Table 5.3 on page 125 improves the result significantly.

The evaluation of an example of the sequence *walking, running, jumping* and *standing* is shown in Figure 6.10. On the one hand, the distinction of *running* and *jumping* is improved by the approaches based on the learnt BN structure, on the other hand, the transition between *walking* and *running* is better recognised in the Naïve Bayes approaches than by the static estimator based on the learnt BN structure. Both situations, distinction of running and jumping as well as the transition between walking and running, are reliably recognised by the dynamic estimator based on the learnt BN structure.

The sequence *walking, falling* and *lying* has been analysed in Figure 6.11. All four approaches work well in general. It is visible however that the duration of falling (at timestamp 166 s on the time axis) is constantly over-estimated, especially by the Naïve Bayes estimators. This situation decreases the precision of *falling* and the recall of the subsequent activity.

The overall recall and precision of the system for every single motion related activity are shown in Figure 6.12. The advantage of the estimators based on the learnt BN structure as opposed to the Naïve Bayes approach can be clearly identified.

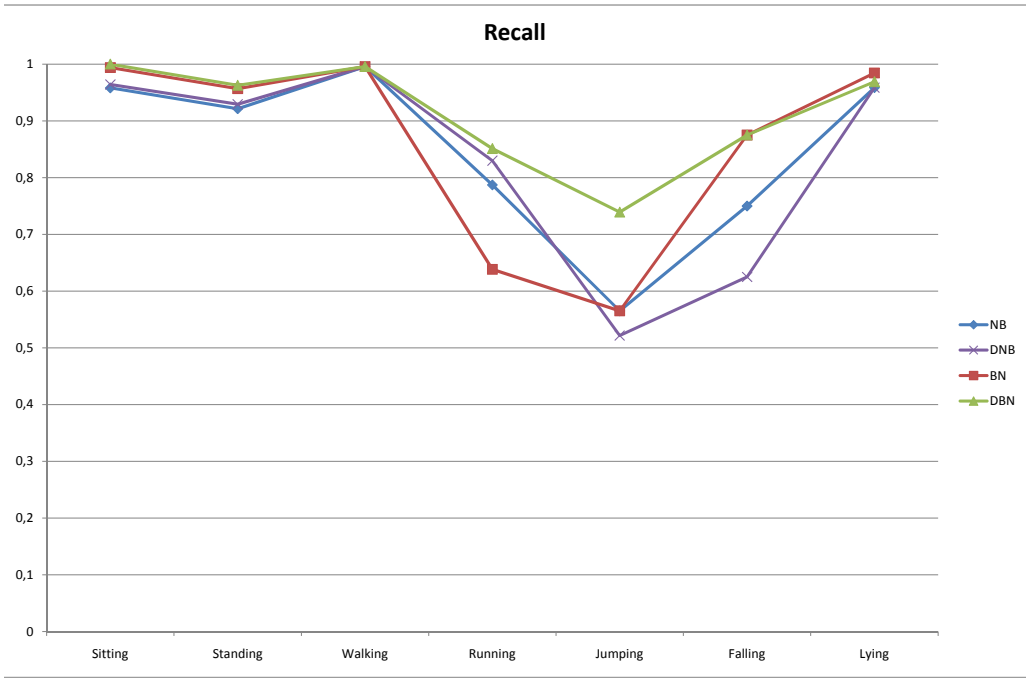
Although the system has been able to recognise every activity at some point of its duration, most activities are misclassified at their beginning, which affects their precision. This recognition delay can be identified in all plots of Figures 6.9, 6.10 and 6.11. As observed above, the over-estimation of the duration in addition reduces the recall. Particularly visible are these effects for short-time activities. For *falling* which lasts for about three to five evidence samples (roughly one second), this has a strong impact and causes the significantly lower numbers for both, precision and recall in Figure 6.12.

Also *jumping* is affected strongly by these effects because of its short duration, but suffers moreover from its division in two phases, getting off and landing, with similarities to *running* and *falling*, respectively.

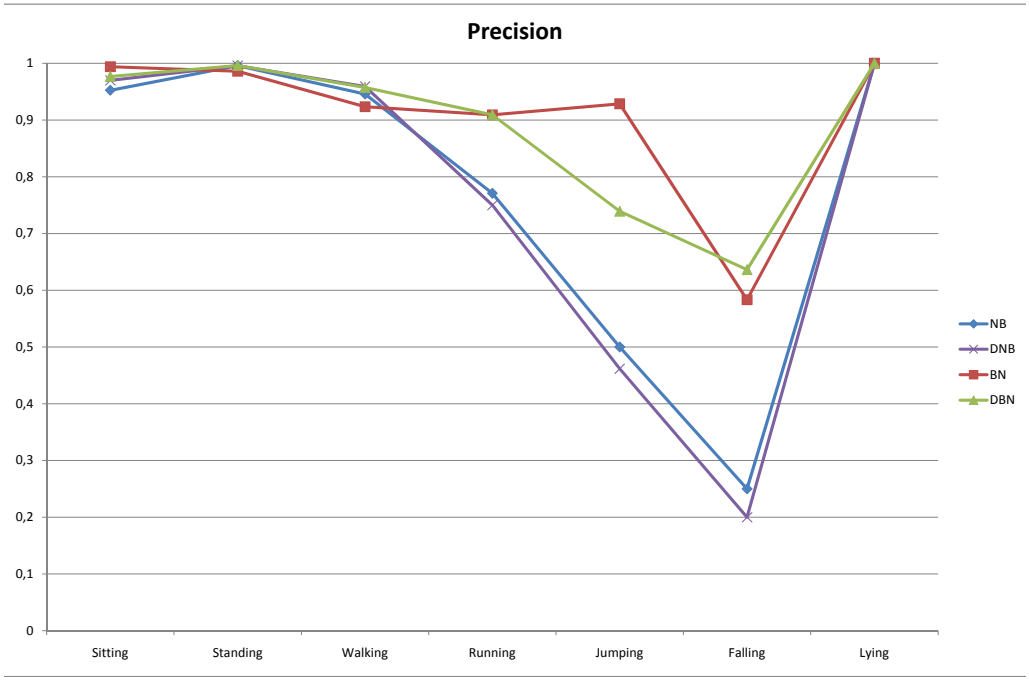
The recognition delay is caused by the sliding windows containing data samples from the previous activity. At an inference rate of 4 Hz, after an activity switch, only 25 samples of a new activity are contained in the evaluated sliding window. The length of the window therefore determines the importance of the last activity in the new classification. The majority of the features presented in section 5.3.2 are mainly based on 128 samples (or 1.28 s). Therefore not until 0.64 s have passed, the majority of samples in a window belong to the current activity.

Taking this into account, a recognition delay of two inference samples, i.e. 0.5 s, is realistic. The recall and precision rates compensating this recognition delay are shown in Table 6.1 for static and dynamic inference based on the learnt BN structure.

Obviously, the dynamic approach yields better recall rates. The dynamic approach also improves precision (particularly for *falling*) in most of the activities except for *jumping* and *sitting*. Taking into account the recognition delay, all activities achieve recall rates higher than 93 %. The remaining errors are partly caused by the manually defined transition model.



(a)



(b)

Figure 6.12: Recall (a) and precision (b) for every activity with the four compared estimators [209]. The approaches using the learnt BN structure outperform the Naïve Bayes classifiers.

Static estimator based on learnt BN structure							
	Sitting	Standing	Walking	Running	Jumping	Falling	Lying
Recall	0.99	0.96	1	0.69	0.66	1	0.99
Precision	0.99	0.98	0.94	1	1	0.57	1
Dynamic estimator based on learnt BN structure							
	Sitting	Standing	Walking	Running	Jumping	Falling	Lying
Recall	1	0.98	1	0.93	0.93	1	0.98
Precision	0.97	1	0.98	1	0.93	0.8	1

Table 6.1: Precision and recall for static and dynamic inference in the Bayesian network with learnt structure [209]. These results compensate a recognition delay of 0.5 s caused by the sliding windows and the inference frequency.

6.2.3 Resource Consumption

All in all, the estimator using the learnt BN structure and the transition model with the grid based filter yields the best results. It is, however, also the most complex estimator, because the learnt network structure is stronger interlinked than the Naïve Bayes approach and the transition model adds further inference complexity. This section investigates, if the qualitative advantage justifies the higher inference complexity by comparing the resource consumption of the different methods.

As described in section 6.2.1 the results shown here are averaged over 780 runs of the Java implementation of the activity recognition system.

Table 6.2 compares the inference time of the different estimators. The average duration of all feature computations is given as reference factor. Feature computation is fast, because all 19 features have been selected also for their low computational complexity. The execution time is specified by its mean, its median, minimum and maximum.

Operation	mean (ms)	median (ms)	min (ms)	max (ms)
Feature computation	1.5	1.45	1.4	4.1
Static estimator, Naïve Bayes	0.34	0.32	0.29	2.17
Dynamic estimator, Naïve Bayes	0.36	0.34	0.3	3.26
Static estimator, learnt BN structure	7.2	7.2	3.9	27.7
Dynamic estimator, learnt BN structure	7.7	7.7	4.1	18

Table 6.2: Execution times of feature computation and inference process from 780 runs on an Intel Core 2 Duo microprocessor E8400 ,at 3.0 GHz with 2 GB RAM. Mean, median, the minimum and the maximum specify the execution times for the feature computation and the four estimators based on Naïve Bayes and the learnt Bayesian network. Computation times for all estimators usually stay below 10 ms and allow for real-time activity recognition.

The computational advantage of Naïve Bayes approaches is significant. Inference only takes from 0.3 ms to 0.4 ms. In contrast, the estimators based on the learnt Bayesian network structure take between 7 ms and 8 ms, hence the twenty-fold. This is due to the complexity of the network (the learnt BN structure has 59 edges, as opposed to the 19 edges in the Naïve Bayes structure, see Figure 5.26) and the resulting bigger CPTs.

Inference incorporating the transition model with the grid-based filter takes around 0.5 ms more than the static approach. Dealing with the HMM increases the inference time, but the main computational cost in terms of execution time is determined by the Bayesian network structure.

This fact is stressed by the analysis of the memory consumption. Table 6.3 shows that also the memory size of the learnt Bayesian network structure and the Naïve Bayes structure differs significantly because of the different complexities of both networks.

BN	Memory size
Naïve Bayes structure	33.3 KB
Learnt BN structure	3706.9 KB

Table 6.3: Memory size of the Bayesian networks after the training process (unrestricted Bayesian network) and for the assumption of Naïve Bayes.

6.2.4 Discussion

The evaluation has shown that activities can be recognised in real-time. Classifying the measurements with the grid based filter based on the learnt BN structure which has proven best in section 6.2.2, the recognition time amounts in total to approximately 10 ms, which allows for classification with 100 Hz, the selected sample rate of the used IMU.

Given that classification with 4 Hz yields excellent results, activity recognition in real-time is even realisable on processors with less resources or running as a background process. Even a reduction of inference frequency would still yield good results (compare [237]) which should be easily achievable also on small mobile devices like smartphones.

The evaluation of precision and recall in Figure 6.12 has shown that the Naïve Bayes approach has significant quality shortcomings which cannot be compensated by its very fast inference time. Neither is the inference time advantage of the static estimator based on the learnt BN structure sufficiently prominent to compensate the somewhat lower precision and recall rates of the recognition.

In the four-fold cross validation, the results have proven to be stable and reliable for users who have not been involved in the training process. Together with the short inference time and the unobtrusiveness of the system with a small sensor module at a single location of the body, this makes the proposed approach an ideal inference module for ubiquitous computing.

6.3 Dynamic Value Ranges

A promising means to reduce the computational demands and therefore also response time of context inference is the concept of dynamic value ranges (DVR). As presented in section 5.4.3, it allows for the reduction of the value ranges of random variables in a Bayeslet according to current requirements and relevancy.

Different criteria have been proposed to control the dynamic reduction and expansion of the value ranges. These are compared in the following sections in an example use case taken from the office scenario presented in section 1.2.1 on page 4. Then the added value of this approach, i.e. the acceleration of inference, is weighted against its increased management costs, following the publications by the author et al. in [119, 117].

6.3.1 Evaluation Settings

Both, the different criteria, as well as the overall utility of the proposed approach are evaluated for an example application, namely the intelligent interactive wall display shown in Figures 1.2 and 1.3 of section 1.2.1. It is situated in the entrance hall of an office building where it shows all employees some information which is relevant for them personally at

the moment when they are standing in front of it, taking into account also their privacy when other people are around.

In particular, it can show a personalised welcome message, information about the canteen menu, the weather forecast, or traffic information. To decide which information is currently most relevant for the display user, his current "work activity" is requested from the context management system. The wall display service has four different general states

1. When the user arrives, he is shown a welcome message, his schedule for the day and the weather prospects (see Figure 1.2).
2. When the user passes by the screen on his way to have lunch, in particular the different canteens' menus are shown (see Figure 1.3).
3. When the user is going home, the traffic information for his way home and possible alternative routes are shown.
4. In other cases, just the tasks and meetings for the current day are shown.

The context necessary for this type of application can be be inferred from information available in office environments already today. The Bayesian network containing the desired context information *WorkActivity* involves the current time, the location of the user and his current computer usage. The computer usage can be measured by light weight daemon services running on the user's computers which monitor the currently used application, the activity time and the last pressed keys. To infer the user's current location, even without a dedicated indoor positioning system, auxiliary information can be acquired from available information. For example a proximity sensor inside the wall display (possibly based on RFID) detects which persons are in proximity. Also the status of the office of a user gives useful information, e.g. if the light is switched on or sensors for ambient noise can give hints about the presence of the user in his office.

These dependencies can be modelled with the five Bayeslets shown in Figure 6.13. However, to be able to better analyse the utility of dynamic value ranges alone, this section considers the monolithic Bayesian network composed of these five Bayeslets, as it could have been designed by a domain expert aware of the available information and sensors in the office building. The example Bayesian network contains hence:

- 17 nodes representing different types and levels of context information. All nodes together initially have 133 different values.
- Seven nodes of them represent sensors or other low level information sources.
- The queried node is *WorkActivity*. It has 18 values representing different activities which are usually pursued during a working day.

From the 18 values of *WorkActivity*, in particular *arriving*, *goingToLunch* and *goingHome* are relevant for the wall display service. These constitute the *states of interest* (see section 5.4.3.3 on page 144), which can be explicitly requested as shown in Figure 5.36. The remaining values of random variable *WorkActivity* are relevant for other services (for instance *Call Redirection* or *Activity Monitoring* from the *Intelligent Office Environment Scenario* in section 1.2.1) and cannot be completely pruned, but their inclusion in inference for the wall display service is not necessary.

The evaluation compares the following state selection criteria (proposed in section 5.4.3.3) to each other and to the reference inference method without reduction of value ranges:

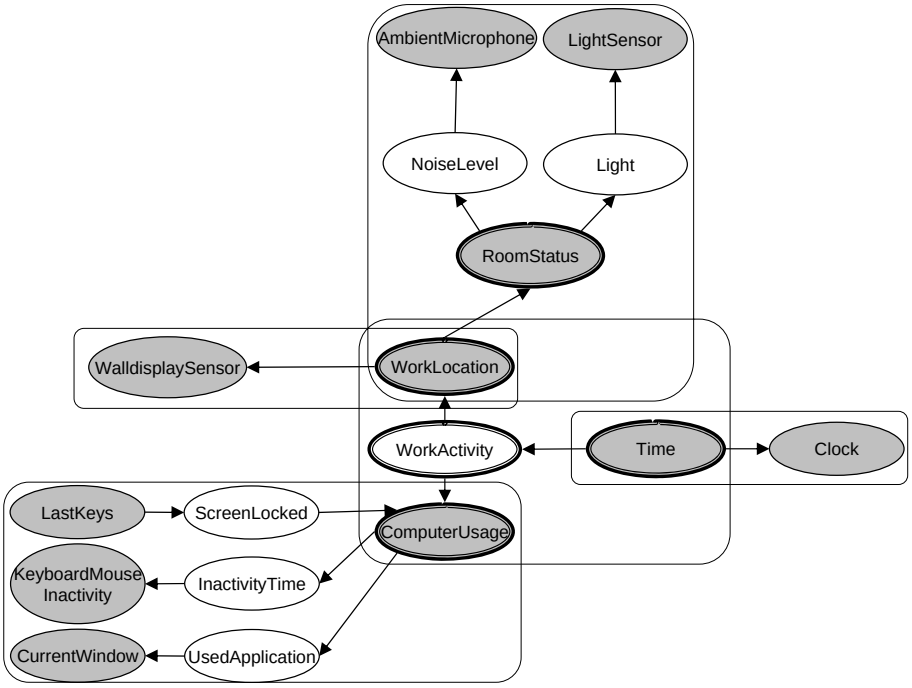


Figure 6.13: Bayesian Network for the inference of 18 different work activities, represented by RV *WorkActivity*, composed of five Bayeslets. A monolithic Bayeslet containing the nodes of all five Bayeslets is used to evaluate the utility of dynamic value ranges in the wall display service of an intelligent office environment, compare section 1.2.1.

- ENS_N, Equal Number of States with N being the number of merged adjacent values.
- MMP, Minimum Merged Probability.
- PE_N, Protection Extension with N being the minimum level of mutual information between a state and its parent or child nodes of interest to be protected. In this approach the above mentioned subset of interest of node *WorkActivity* was chosen and protected. This protection is extended recursively to its parents and children and from them to others along the network. All unprotected states are merged.

The *Tree Collapsing* criterion is not applied here, because its prerequisite, an hierarchically organised value range, is not met. The *States of Interest* have been omitted in the analysis, as its core idea is realised in the *Protection Extension* criteria and it can be approximated with high values of N in PE_N. *Entropy Minimisation Discretisation* was not selected, as its binary discretisation does not promise to provide good results and is only an option for the merging of unprotected states in combination with the *Protection Extension* criterion. The values for N in ENS_N and PE_N have been chosen after an inspection of first results, to reach the best quality of a criterion.

Evaluations were conducted on a notebook with Intel Core 2 Duo T9600 processor at 2.8 GHz with 3 GB of RAM under Windows XP, using the author's Java implementation of the PPTC algorithm and the value range reduction algorithms. The Bayesian network (together with the initial junction tree) are maintained on the object heap of the Java Virtual Machine.

The evaluation results below are based on inference requests propagated through a simplified context management system. The evaluation scenario assumes randomly chosen

hard evidence in the seven nodes classified as input nodes in Figure 6.13. Fifteen realistic configurations of evidence values for the input nodes have been chosen from the 290, 304 possibilities.

In the case of the reference inference with PPTC for the complete value ranges, the evidence is introduced and propagated. For the different state selection criteria, first the value ranges are partitioned according to the selected criteria, the junction tree is rebuilt based on the reduced value ranges and used then to propagate the introduced evidence.

6.3.2 Error Introduced by the Value Range Reduction

To analyse the error introduced by the value range reduction, the posterior probabilities of the output RV *WorkActivity* by inference in the original BN were compared with the inference results of the BNs partitioned with a method m. Inference was based on 15 evidence cases, each specifying hard evidence for all input nodes.

The measure for the inference error of a value v of RV V is the *mean absolute error*:

$$MAE(v) = \frac{1}{N} \sum_{i=1}^{N} |P(V = v|\mathbf{E} = \mathbf{e}_i) - P(V_m = v|\mathbf{E_m} = \mathbf{e}_i)| \ , \tag{6.3}$$

where $N = 15$ is the number of evidence cases $\mathbf{e}_i$ considered. A RV X_m is derived from RV X by partitioning its value range with method m. Analogously, a set of RVs $\mathbf{Y}_m$ is derived from set $\mathbf{Y}$ by partitioning every node in the set with method m.

In this measure, all considered evidence cases are weighted equally and the average deviation from the posterior probability inferred with exact inference is calculated. For example, if $P(arriving|\mathbf{e}_i) = 0.90$ and $P(WorkActivity_m = arriving|\mathbf{E_m} = \mathbf{e}_i) = 0.9029$, then $MAE(arriving) = 0.0029 = 0.29\%$.

For those values v_j of *WorkActivity* which have been merged by a method m to a super-state $\widehat{\mathbf{v}} = \bigcup v_j$, their conditional probability $P(V_m = v_j|\mathbf{E_m} = \mathbf{e}_i)$ is calculated from $P(V_m = \widehat{\mathbf{v}}|\mathbf{E_m} = \mathbf{e}_i)$ according to the relations of the prior probabilities of the values v_j.

Table 6.4 shows the mean absolute errors of all values of *WorkActivity* for the considered evidence cases. It can be seen that the PE methods perform significantly better than ENS_4 and MMP for the protected states of interest specified by the service. Their errors are increased for values out of the interest range, but still acceptable. There is only a small difference between the stronger protection extension ($PE_{0.01}$) and the weaker one. Both approaches however imply in total larger value ranges than the remaining methods.

Figure 6.14 focusses on the three states of interest, *arriving*, *goingToLunch* and *goingHome*, showing the mean posterior probabilities for the four different approaches to DVRs and the posterior probabilities in the original BN without dynamic value ranges. It is obvious that the best performance for the states of interest is provided by the protection extension methods.

It can also be seen that apart from the protected states, the error increases immediately for the PE methods, as all unprotected states are merged and the evaluated posterior probabilities only reflect the prior probabilities. In certain evidence cases, the error hence can be much higher. This error in the unprotected states however is irrelevant for the quality of the PE methods, as they are not of interest for the requesting service.

	ENS_4	MMP	$PE_{0.03}$	$PE_{0.01}$
#states	42	64	74	89
ARV	2.75	3.38	0.62	0.29
GTL	5.99	5.85	1.09	0.89
GH	12.94	13.05	1.63	1.63
PRG	9.34	12.61	6.07	6.14
RDG	5.56	4.27	4.53	4.45
WRT	6.03	2.15	6.93	6.71
WLK	3.54	2.43	2.55	2.57
SLP	1.59	1.13	1.03	1.04
ENT	0.30	0.24	0.25	0.25
RLX	0.17	0.11	0.12	0.12
TLK	1.28	0.69	0.67	0.72
THK	0.34	0.88	0.89	0.88
LST	1.08	1.15	1.17	1.16
PRS	1.10	1.71	1.25	1.24
LNC	0.94	0.42	0.91	0.92
DRV	1.53	1.18	1.47	1.48
SPR	0.29	0.39	0.52	0.53
OTH	0.34	0.32	0.59	0.59

Table 6.4: Mean absolute error (in %) of the posterior probabilities for all state selection criteria and all values (left column) of the output node *WorkActivity* in 15 cases assigning hard evidence to all input nodes defined in Figure 6.13. The abbreviations ARV,GTL and GH stand for the values *arriving*, *goingToLunch* and *goingHome* respectively. The second row gives the total number of values in the Bayesian network after processing all value ranges with the four different state selection criteria.

6.3.3 Resource Consumption

The quality of a criterion is determined by the advantage in terms of reduced resource consumption and the error introduced by the deviation from exact inference with complete value ranges. The resource reduction which is important for light-weight mobile devices refers to both, memory and CPU time.

The memory requirements are influenced by the need for additional data structures for the dynamic value ranges (among others maintenance of the original value ranges) and by the reduced sizes of the potentials of the clusters in PPTC. The exact consequences have been evaluated by the author et al. in [119], for both the execution memory and the permanent storage of a representation of the reduced value ranges. As the execution memory for inference with PPTC however does not differ a lot, this thesis refrains from detailing on the memory consumption.

The analysis presented in Figure 6.15 compares the duration of the complete context inference process after a first inference request, assuming that the Bayeslet and its junction tree are maintained already in the execution memory of the context management system. A context inference engine following the process presented in section 5.4.3 first has to reduce the value ranges and adapt the junction tree, second to introduce the available evidence and third and last propagate the evidence to calculate the posterior probabilities, i.e. the actual inference. These three steps of context inference together with the overall execution time are shown in the figure for the four compared repartitioning methods and for inference in the original Bayesian network, where the repartitioning step is left out.

The most obvious fact is the significant reduction of the core inference task, the propa-

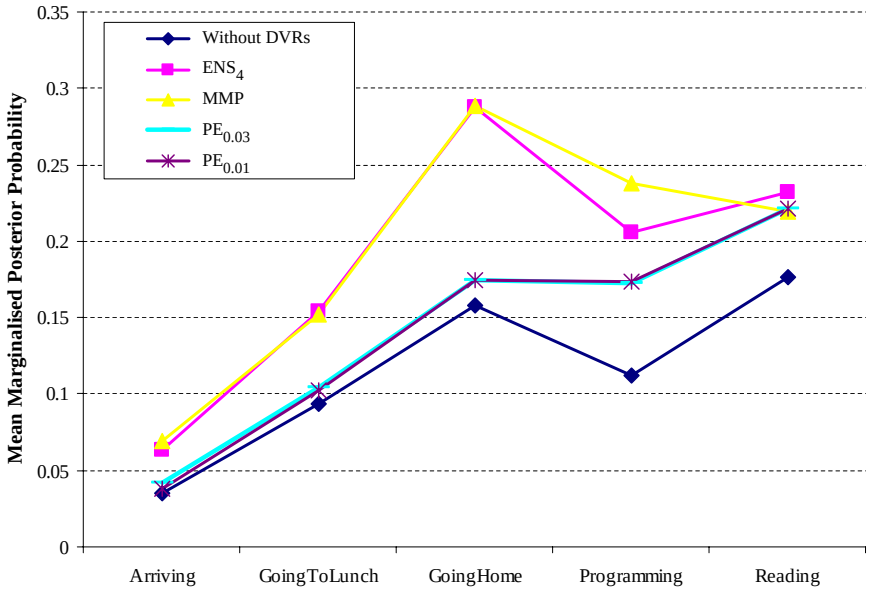


Figure 6.14: Mean inferred posterior probabilities of five values of the node *WorkActivity* for 15 defined evidence cases. The PE methods yield results very close to those of the original network for the states of interest *arriving, goingToLunch* and *goingHome*, while its error increases for the unprotected values *programming* and *reading*. The partitioning methods ENS_4 and MMP differ stronger from the original values, but also follow the overall trend of the original posterior probabilities.

	$NoRep$	ENS_4	MMP	$PE_{0.03}$	$PE_{0.01}$
total no. values	133	42	64	74	89
inference time	2.43	0.23	0.39	0.41	0.58

Table 6.5: Probability propagation time in ms, in relation to the total number of values in the BN.

gation of probabilities through the BN. This step is most impacted by the complexity of a BN, as it has to run through the complete potentials of the clusters. Therefore it benefits most from the reduced value ranges. Table 6.5 shows that inference in the most reduced BN (with ENS_4) needs only a tenth of the time of inference in the full BN. And even the most complex repartitioning method, $PE_{0.01}$, needs less than a fourth of the time with two thirds of the original values.

The tiny differences for evidence introduction to the advantage of DVRs are caused by the smaller value ranges. This could however be avoided by a more efficient implementation. The biggest time factor for inference with DVRs however is the reduction of the value ranges itself. While repartitioning for ENS_4 and MMP with around 1 ms is still short enough to have an overall inference time below the overall inference time without DVRs, repartitioning with the PE method takes around 6 ms. The increased effort however yields the good inference results for the protected states with only minimum error, as has been seen in Table 6.4 and Figure 6.14.

As the repartitioning step does not have to be run every time but only once, the advantage of using repartitioning from the second inference request onwards is obvious. This development is shown in Figures 6.16 and 6.17. Figure 6.16 shows in detail that from the fourth inference request on also the PE methods have amortised. Figure 6.17 extends the development to more than 60 requests – which would be the situation of continuous inference with 1 Hz.

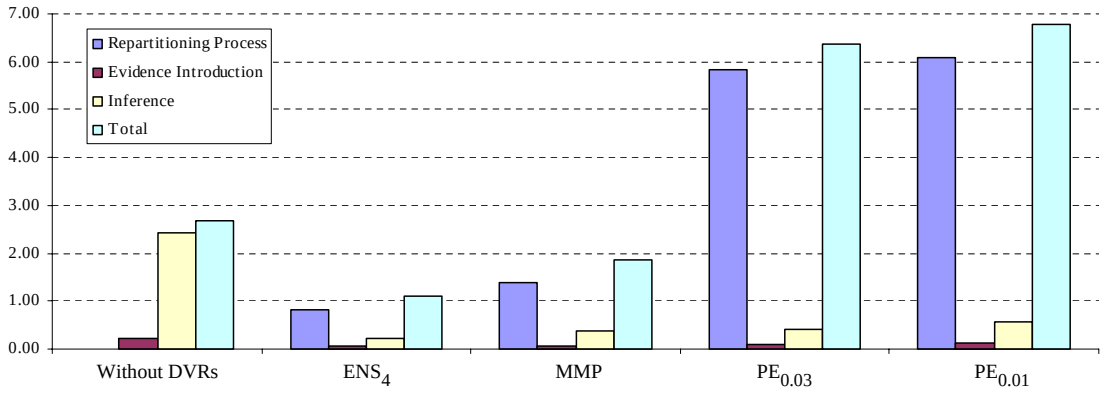


Figure 6.15: Comparison of inference duration (in ms) with and without dynamic value ranges considering four different approaches for reducing the value ranges. Context inference is divided into three steps: partitioning the value ranges, introducing evidence and propagating the probabilities through the network.

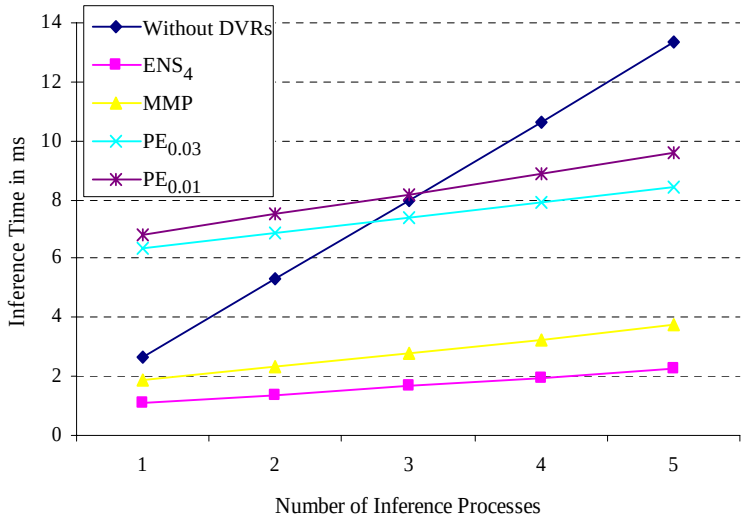


Figure 6.16: Development of inference time with the number of executions for the first five inference requests, illustrating better the "break even" of the methods for dynamic value ranges.

As many services are used repeatedly, the graph of Figure 6.17 is more relevant than the single-inference graph of Figure 6.15. If we assume that employees pass by the foyer about 10 times a day, and this happens every working day, a significant part of inference time can be saved. Every week, at least 75 % of the inference resource consumption for context inference of this service would be saved.

Although the gain per execution is only a couple of milliseconds, this is relevant indeed when several inference processes are running in parallel. According to the projection from chapter 3, pp. 51 et seqq., about seven context attributes will be inferred continuously in parallel, not to mention further on-demand context requests. With plenty of running services on a resource constrained mobile device, also such little savings sum up and are relevant for reducing response time and, in addition, save battery life.

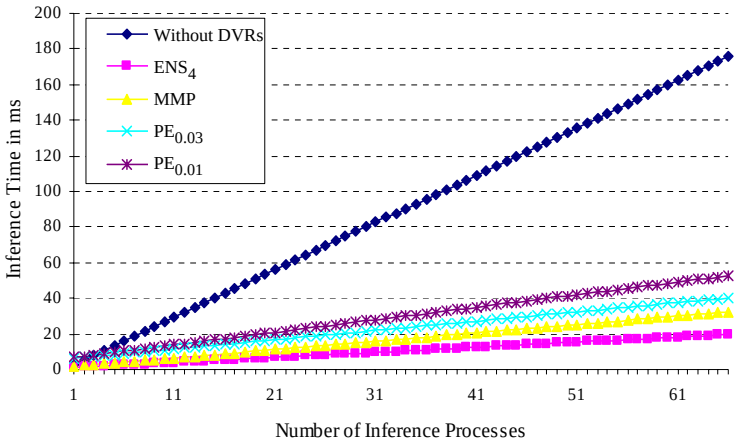


Figure 6.17: Further development of inference times with higher numbers of executions, generalising from the initial scenario shown in Figure 6.16.

6.3.4 Discussion

The evaluation of this application example shows that dynamic value ranges can reduce inference time considerably introducing only small errors to the inferred posterior probabilities.

This result can be generalised: The time necessary for calculating the reduced value ranges will not change significantly, as it depends on the size of the BN. Having assumed earlier an average Bayeslet size of 20 – 30 nodes, this coincides roughly with the size of our example. The gain of probability calculation in the BN depends on the inference methods which largely maintain their reduction rates. Hence the relative savings in terms of CPU time should be quite stable.

The approach of DVRs yields its advantages mainly for general context inference without additional assumptions (e.g. on the network structure or evidence distribution). In this case the full conditional probability tables which depend strongly on the size of the value ranges have highest relevance. A Bayeslet like the one for human motion related activity recognition for instance, where it is known that all nodes but the output node are input nodes, other inference techniques are efficient and the more complex, general approaches such as PPTC can be avoided.

With regards to the proposed approaches for value range reduction, all methods have produced acceptable results in terms of their mean absolute errors. The approaches based on protecting states of interest and extending this protection to related nodes thereby yield the best results, with negligible errors for the states of interest.

Although the time for the value range reduction in these approaches is significantly higher than for other methods and even exceeds standard inference duration without DVRs, they are the most appropriate choice. Their disadvantage for the value range reduction can be neglected, as it happens only once while usually the requesting services are executed frequently or even continuously.

Moreover, this approach benefits from taking into account application knowledge, as the requesting service specifies the states of interest. Similarly to focussing on specific output and input nodes which allows for modularisation of BNs, the focus on concrete states of interest allows for the reduction of inference complexity to the relevant parts.

6.4 Composition of Bayeslets

Section 5.4.4 has proposed a methodology to select the Bayeslets to be connected. The relevancy of a Bayeslets thereby is determined with its expected utility for the Bayeslet it could be connected to. The following subsections shall verify both utility measures proposed in section 5.4.4.2 on pp. 147 et seqq.

First, section 6.4.1 presents the common application scenario. Section 6.4.2 applies the probability based utility function to this scenario, before section 6.4.3 presents a solution with the decision based utility. Finally, the advantages and shortcomings of both approaches are discussed in section 6.4.4.

6.4.1 Application Example

In the following the decision algorithm introduced in the previous sections will be shown by the exemplary application Cooperative Adaptive Cruise Control (CACC) [103, 46]. This application is used in the scenario from section 1.2.2, pp. 6 et seqq.

More and more, cars support vehicle-to-vehicle (V2V) communications, as developed for instance by the CAR 2 CAR Communication Consortium[3]. V2V communications are used to provide enhanced driver assistance applications by taking into account information (e.g. sensor measurements) received from other vehicles.

Hence, the vehicles are part of the ubiquitous computing environment. Since the communication channel has to be used collaboratively by all vehicles in the network and timely reception of messages is crucial for safety relevant applications like CACC, bandwidth has to be shared intelligently. This implies in particular that not relevant information should not be transferred.

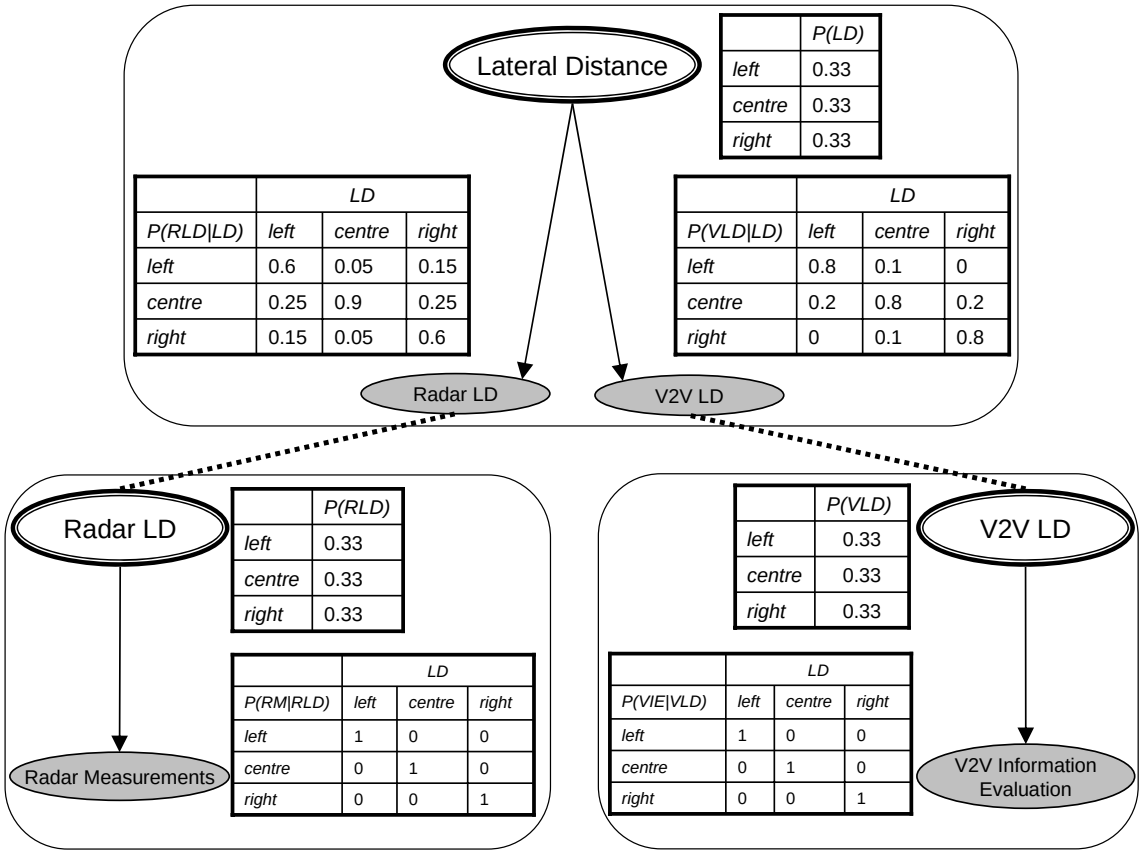
Applied to context inference, this means that a vehicle trying to infer its status should not request soft evidence from a remote Bayeslet which does not add significant value. Transmission over the wireless channel would consume bandwidth and thereby prevent others from transmitting their, maybe more important, information.

In the following, this thesis considers a CACC application running in a car. On a road with multiple lanes in the same direction CACC has to be aware of the actual lane the preceding vehicle is driving on. If, for instance, the car is located on the centre lane of a three-laned road, the preceding vehicle can be located on the same, the right or the left lane. Context inference can infer this from the available information and provide this information to CACC [123].

A part of this situation is modelled in Figure 6.18. The three states of the possible location of the preceding vehicle are denoted in the following as values *centre*, *right* and *left* of the random variable *Lateral Distance (LD)*. Information sources for the lateral distance are the built-in radar system of the car itself, but also the messages with GPS information of the preceding vehicle received via V2V communications. Both information sources form Bayeslets which can be connected to the Bayeslet for the *Lateral Distance*, if their $NetEU$, cf. Eq.(5.73), is sufficiently high.

The Bayeslets used for this purpose are simplified to the minimal case so the concepts from section 5.4.4.2, pp. 147 et seqq., can be illustrated best. In particular the Bayeslets of the information sources have been strongly simplified, so they have the same priors of their output nodes, the same size, and the same connections. Both information source Bayeslets hence have the same computational costs, so the evaluation can focus on the two proposed utility functions.

[3] `http://www.car-to-car.org/`



	P(LD)
left	0.33
centre	0.33
right	0.33

	LD		
P(RLD|LD)	left	centre	right
left	0.6	0.05	0.15
centre	0.25	0.9	0.25
right	0.15	0.05	0.6

	LD		
P(VLD|LD)	left	centre	right
left	0.8	0.1	0
centre	0.2	0.8	0.2
right	0	0.1	0.8

	P(RLD)
left	0.33
centre	0.33
right	0.33

	LD		
P(RM|RLD)	left	centre	right
left	1	0	0
centre	0	1	0
right	0	0	1

	P(VLD)
left	0.33
centre	0.33
right	0.33

	LD		
P(VIE|VLD)	left	centre	right
left	1	0	0
centre	0	1	0
right	0	0	1

Figure 6.18: Three (simplified) Bayeslets to infer the *Lateral Distance* of two cars for *Context Aware Adaptive Cruise Control*. The dotted lines indicate the possible connections of the Bayeslets, the boxes next to the random variables give their corresponding prior and conditional probability tables, respectively.

6.4.2 Mutual Information Based Composition Decision

Section 5.4.4.2 has presented the mutual information $I(\mathbf{X} : \mathbf{Y})$ as an appropriate, probability based utility function to evaluate the relevancy of a Bayeslet $\mathbf{Y}$ for the evaluation of Bayeslet $\mathbf{X}$. Thereby, the mutual information between two Bayeslets $\mathbf{X}$ and $\mathbf{Y}$ is reduced to calculating the mutual information between two random variables in the same Bayesian network – the output node of $\mathbf{X}$ and the input node of $\mathbf{X}$ with the prior probabilities of $\mathbf{Y}$ as evidence.

The following evaluation concentrates on the behaviour of the mutual information between the three Bayeslets (**LD**, **RLD**, **VLD**) shown in Figure 6.18, as the cost function for both input Bayeslets are the same, following the proposed cost function for computational costs from Eq.(5.77). $H(\mathbf{RLD}) = H(\mathbf{VLD}) = 0.86$, as the output variables of both Bayeslets have three values with uniformly distributed prior probability. The same size of the Bayeslets results then in the identical value for the cost function, e.g. $C(\mathbf{RLD}) = C(\mathbf{VLD}) = 0.77$, if $a = b = 1$.

In the simplistic example used, Bayeslet **LD** has to decide, if one or both of the Bayeslets **RLD** and **VLD** should be connected. Bayeslet **LD** only contains input and output nodes, so the decision is made without incorporating any evidence:

$$I(\mathbf{LD} : \mathbf{RLD}) = 0.43 \tag{6.4}$$

$$I(\mathbf{LD} : \mathbf{VLD}) = 0.78 \tag{6.5}$$

Both Bayeslets hence add value to Bayeslet **LD**. The value of Bayeslet **VLD**, however, is expected to be higher, as the conditional probability distribution of node VLD expresses higher certainty and the CPT of RLD expresses lower sensor quality for the detection of $left$ and $right$, e.g. due to reflections on the guard rails. Neglecting transmission costs, the Bayeslet **VLD** would hence be selected.

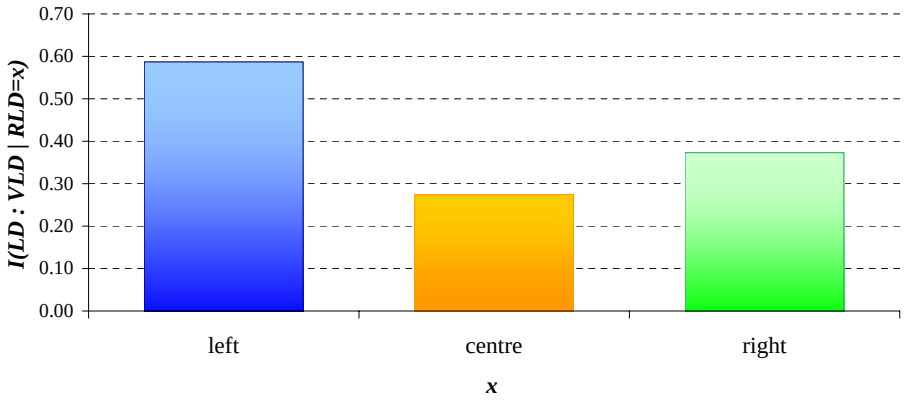


Figure 6.19: Evaluation of the mutual information between the Bayeslets LD and VLD given the different outcomes of Bayeslet RLD.

In the case of cars, the computing power is not as limited as for small devices. The critical resource in traffic is rather the network bandwidth. Taking this into account, context inference could decide in this case to always connect all local sensor systems like radar and to evaluate only the relevancy of external sources like the Bayeslet **VLD**. Its relevancy depends on the values observed for the other Bayeslets.

Figure 6.19 shows the results for the mutual information between the Bayeslets **LD** and **VLD** given all possible observed values of Bayeslet **RLD**. The mutual information varies between 0.59 and 0.27, as the utility of an additional Bayeslet depends on the probability of LD which has already been influenced by the already available evidence. The different degrees to which it still can be impacted by the additional Bayeslet determine the mutual information. Depending on the cost function for the transmission costs, Bayeslet **VLD** would hence be connected in some cases, in others not.

6.4.3 Value of Information Based Composition Decision

As explained in section 5.4.4.3 and shown in Figure 5.39 on page 151, also the decision based utility function can be used to determine the (un-)certainty introduced by a Bayeslet. However, it also offers the possibility that a Bayeslet creator defines more specific utility functions and decisions. This thesis demonstrates the latter option, according to the work of the author, Röckl and Pfeifer in [123]:

Here, a tripartite utility function has been specified. The basic utility functions measure the *Safety* and the *Efficiency* utility of an action. The overall utility function then represents the weighted sum of the two basic ones at the ratio of 3:1. The decision differentiates the two states *accelerate* and *decelerate*. The whole probabilistic decision network, extending the situation shown in Figure 6.18, is depicted in Figure 6.20. The explicit representation of the Bayeslets is neglected here as their outcome is introduced as evidence into the nodes *Radar LD* and *V2V LD* and the value of information can be computed only in this Bayeslet.

If no Bayeslet provides information, the state of LD is uniformly distributed, each state with a probability of 1/3, see Figure 6.21 (left). In this case there is a tie between

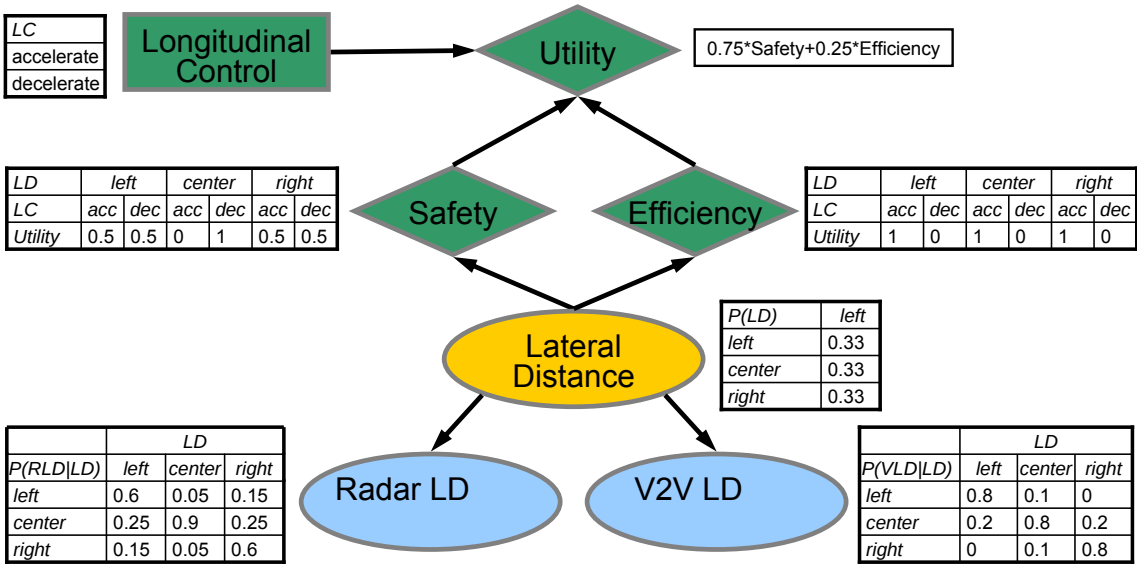


Figure 6.20: *Longitudinal Control* decision based on the *Lateral Distance* considering the information sources radar and V2V communication like in Figure 6.18. *Efficiency* and *Safety* measure the utilities as shown in the boxes next to their node representations. The overall utility influencing the decision is calculated by $0.75 \cdot \text{Safety} + 0.25 \cdot \text{Efficiency}$. Taken from [123].

the actions *accelerate* and *decelerate*. Both have a utility of 0.5. In order to decide whether another Bayeslet including additional sensor measurements either from radar or V2V communication should be connected, the VoI is calculated. Figure 6.21 (left) shows that both Bayeslets yield $VoI > 0$, hence are relevant. $VoI(RLD) = 0.11$ and thus carries more information than the Bayeslet for the information from V2V communications with $VoI(VLD) = 0.10$. These calculations are independent of the unknown actual state of the evidence and only use the prior probabilities of the Bayeslets' output nodes available from their specifications. Thus, the Bayeslet of the radar measurements is connected and evaluated.

When the radar Bayeslet is connected and has provided evidence to the input node RLD the value of information and therefore the decision whether to connect the second Bayeslet is different. If, for instance, the radar has measured a *centre* state, the VoI of VLD reduces to 0.07, because both evidences are dependent due to the common cause LD. Thus, the belief in the *centre* state as the actual outcome of the VLD already has a probability of 59 %, see Figure 6.22 (middle).

The VoI of VLD is still positive however and, hence, is expected to add value for decision-making. This is justified, as the MEU can still change significantly, depending on the actual value of the second Bayeslet. For example, after the incorporation of the state *centre* for RLD a connection of the second Bayeslet to VLD may provide state *left*, as depicted in Figure 6.21 (right). This makes the MEU change to 0.51 for the *accelerate* action. Thus, the best action changes from *decelerate* to *accelerate* by the connection of the second Bayeslet.

Obviously, when transmission costs are neglected in this example, position information from the preceding vehicle via V2V communications may provide a valuable benefit for decision-making.

The VoI is not static for Bayeslets, but depends on the evidence already available. This can be seen in Figure 6.22. The figure is based on the same parameters and assumptions as

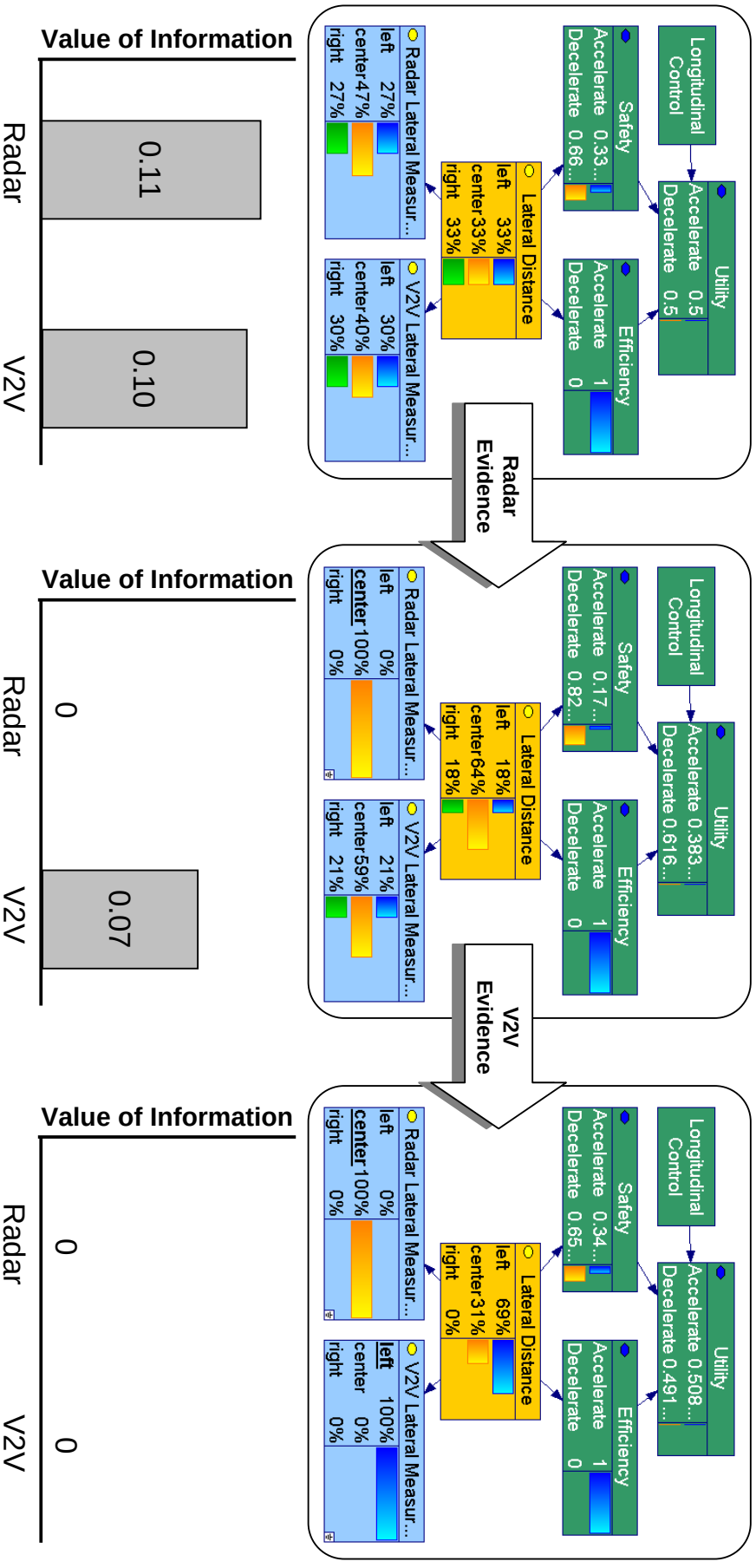


Figure 6.21: Variation of the value of information for different connection states of the Bayeslets for radar and V2V communications. Taken from [123].

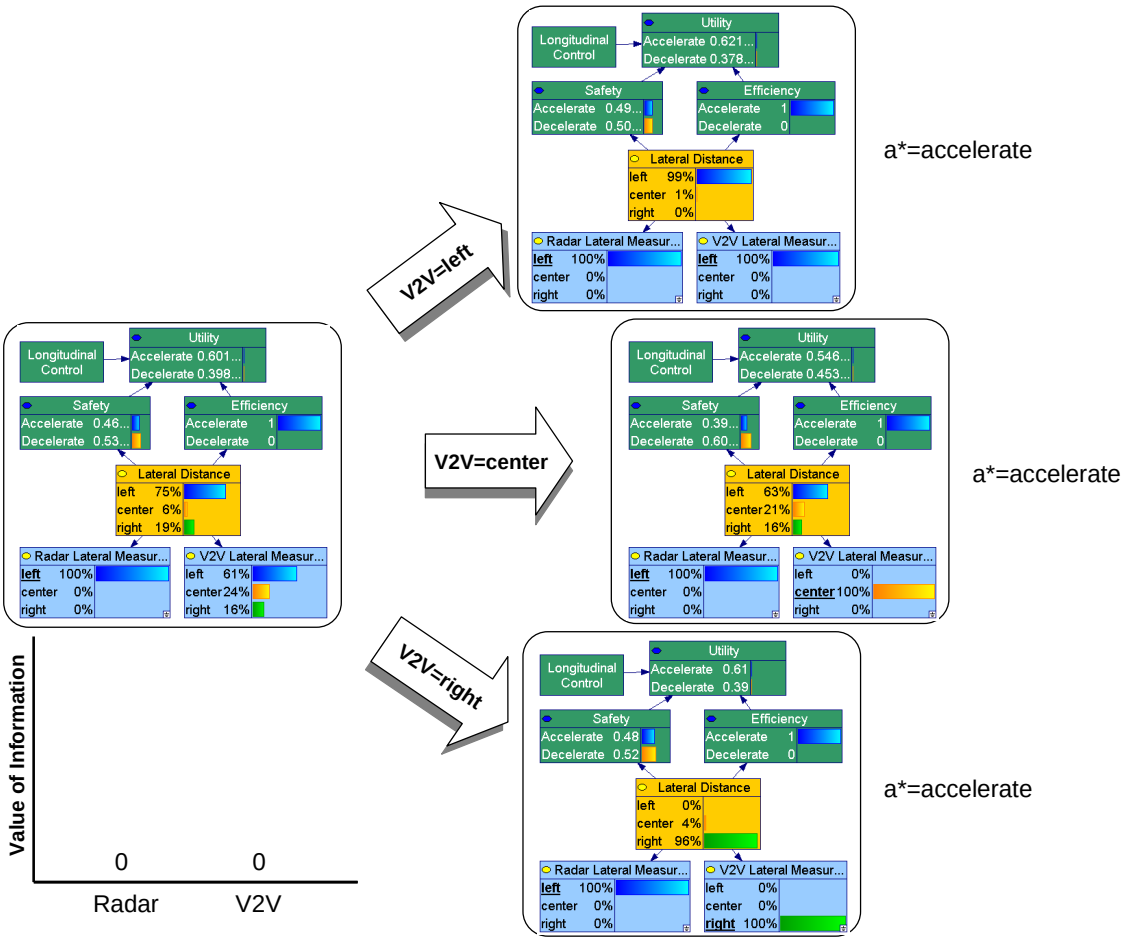


Figure 6.22: Variation of the maximum expected utility (MEU) with different outcomes of the V2V evidence after observing the radar evidence *left*. Taken from [123].

Figure 6.21, but instead of *centre*, the outcome of the Bayeslet for *RLD* is *left*. Accordingly, the best action is *accelerate* with a MEU of 0.60. In this case $VoI(VLD) = 0$.

The figure shows on the right side the consequences of a hypothetical connection of the Bayeslet for *VLD*. If its evaluation yields *left*, the best action obviously stays *accelerate*. If it provides *centre*, still *accelerate* is the best action and even if it provides *right*, the best action is *accelerate*. Thus, independently of the outcome of the Bayeslet the decision remains unchanged. Hence, it does not add information and this Bayeslet is not connected, saving computational load, transmission time and bandwidth.

6.4.4 Discussion

The last sections have shown how both composition criteria can be applied for a context inference goal. The approaches have shown to provide the desired results, i.e. a comprehensible decision about which Bayeslet is more relevant, taking into account the evidence already available from different sources.

Both approaches yield different results for the used example. Mutual information rates Bayeslet VLD more relevant, value of information is higher for RLD. This is due to the different objectives of both approaches. The objective of mutual information based on

the entropy is always the reduction of uncertainty. The objective of the decision based approach, however, can be defined freely by the configuration of the utility node(s).

Consequently, both methods are not mutually exclusive, but are valid alternatives which can be chosen depending on the current situation. The creator/owner of a Bayeslet hence has the option to define decision and utility nodes for a specific objective. If no objective is defined, the standard would be the uncertainty reduction. This can be realised with the mutual information based composition decision.

The final performance of both approaches also depends on the cost function. The cost function defined in Eq.(5.77) works fine for determining the computational cost of a Bayeslet. However it has to be complemented with a measure for the transmission costs which would depend on the network status and the needs of the application.

6.5 Tractability Evaluation

In chapter 3 context usage of ubiquitous computing has been quantified with the conclusion that a central approach would be intractable. Exact Bayesian inference on resource constrained, mobile devices however has narrow limits due to the NP-hardness of Bayesian inference, see section 2.3. To fulfil the resulting requirements, (i) decentralised, modular inference, (ii) reduction to relevant input, (iii) demand driven inference, (iv) support for different characteristics of context, (v) access control and privacy, and (vi) adaptability, chapter 5 has proposed a set of design patterns and processing approaches for context inference with Bayesian techniques.

The previous sections 6.1 – 6.4 have shown that particular, single Bayeslets can be evaluated efficiently, that their inference time can be further reduced with dynamic value ranges and that there are means to select only relevant Bayeslets for an inference goal. Therefore this section mainly focusses on the evaluation of the modular approach with Bayeslets, linked by soft evidence. An example, presented in the first subsection, shall demonstrate how modularisation of a big Bayesian network improves tractability of inference, both, in terms of inference time (section 6.5.2) and network load (section 6.5.3).

6.5.1 Evaluation Example

The evaluation example in this section is meant to describe large parts of the situation of a user. Not surprisingly, all different parts of a current situation are more or less closely related and influence each other, forming a large Bayesian network shown in Plate A.1 of Appendix A which shall be used in the following evaluation. It is formed from all Bayesian networks used in the preceding sections and some further information which either extends, connects or complements the already shown parts of a user's situation model.

The resulting *Situation* network includes 180 random variables, some with discrete, others with continuous value ranges, connected by 275 edges. It can be grouped into more than 50 Bayeslets in which 80 RVs have been marked as input nodes, i.e. have direct input from sensors or other information sources. As a network with such dimensions is hard to represent on paper, Figure 6.23 shows a simplification of it, where nodes have been grouped into rough categories.

Thereby this network is not complete by far. Modelling of properties of the situation, personal preferences and influence factors could be continued ad infinitum. For instance health related features such as allergies have not been considered in the current example.

The dimensions of the example network have also been limited as to the modelling of domains related to the BN's owner. It considers only one home, one car, one meeting, and two active services, while in reality often more than one "home" is relevant (e.g. the

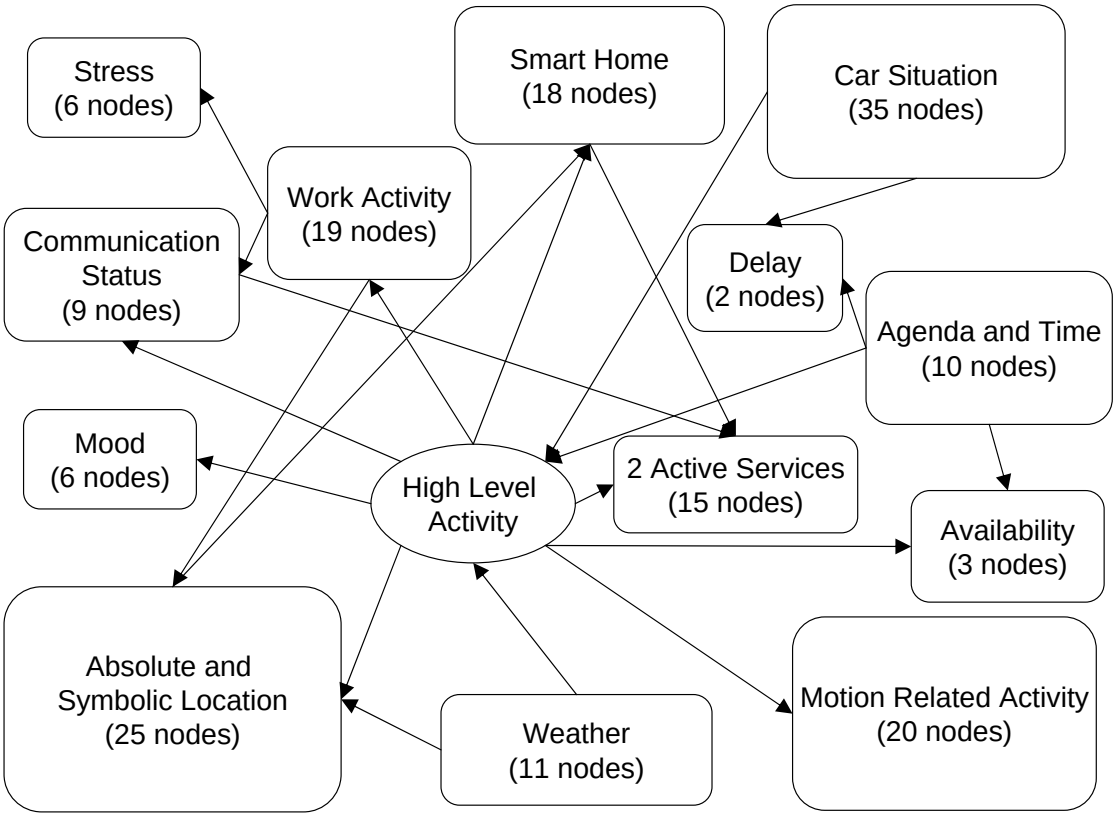


Figure 6.23: Compressed representation of a Bayesian network describing parts of the *Situation* of a user, including the activities at work, location, motion related activity, the situation in his smart home, and the traffic situation of his car. The boxes represent groups of random variables, the arcs between them summarise the connections of the contained RVs. See Plate A.1 for details.

own home, the partner's home, the parents' home, the hotel room, etc.), more than one car is available and, referring back to the scenarios of section 1.2, pp. 3 et seqq., and the context usage analysis in section 3.1 (see for instance Table 3.1 on page 55), more than two services are used in parallel.

Other cars which determine the current traffic situation and hence should be connected to the car's situation are not modelled either. Most importantly however, no other people are referred to in the current model, although their input would be important. Information about other people is referred to by eight of the modelled nodes:

- *High Level Activity*: Often, the activity of people nearby influences the own activity.
- *Availability*: The decision whether a user is available for a communication request depends not last on the person who requests the communication, in particular the relation between the persons and the situation of the communication requester.
- *Car*: A car can be used by more than one person. As the car's situation influences the status of all passengers, knowledge about a co-passenger influences the user's situation.
- Twice in *Smart Home*: Two services modelled for the smart home take into account its residents' location and activities. Just as for the car, the situation of cohabitants influences the user's situation.

- *Meeting*: The location, presence and role of other people (colleagues, conference participants, etc.) gives important information about the status of a meeting.
- Twice at *Work Activity*: Beyond meetings, the daily work routines are influenced by the colleagues. Knowledge about their current activities gives hints about the own activity, knowledge about current communications helps to infer the situation and situations (such as going to lunch) which are usually executed in a group are recognised more easily.

In the most naïve case, the same nodes would have to be connected for every relevant user. And the number of persons possibly relevant is immense, thinking of all persons working at the same office building or campus, going to the same canteen, super market and gym, one's family, friends and acquaintances. An approximation of the most relevant ones can be given by the help of "facebook"[4] which is representative with its currently more than 500 million active users. On average, every user has marked 130 other users as friends. Keeping up all limitations mentioned before, and incorporating the 130 facebook-friends would result in more than 22,000 random variables in the inference network.

The example clearly shows that some modularisation is necessary and limits have to be set for a situation model.

6.5.2 Evaluation Time

To evaluate the benefit of a modular approach in terms of inference time, the monolithic situation BN presented in the previous section is partitioned into four subnets, which can be seen in Figure 6.24.

- Network $A1$ contains the random variables concerning the *Smart Home* and the *Work Activity* which has been described in section 6.3.1 on page 174, extended by nodes for a business meeting. Moreover nodes for *Active Services*, different *Symbolic Locations*, as well as to *Agenda and Time*, *Delay, Mood* and *High Level Activity* are used as interface nodes to the other BNs. The core of this network can be remotely evaluated at servers of the smart home and the workplace.
- Network $A2$ is formed mainly by a static implementation of the dynamic network for *Location* fusion and inference used in section 5.2.2, pp. 88 et seqq., and its extension to symbolic location. In addition, the *Communication Status*, *Active Services* and the *Weather* are parts of the inference network. Additional nodes, used as interface to the other BNs are two nodes providing input about the *Work Activity*, *Delay, Mood* and *High Level Activity*. The information necessary therefore is coming mainly from sensors in the user's local smart space and would be evaluated locally.
- Network $B1$ represents mainly the *Car Situation* (including the lateral distance information presented in section 6.4.1, *Agenda and Time* and *Delay*, as well as interface nodes to *Availability* and *High Level Activity*. The largest part, the *Car Situation* refers to remote information and would be evaluated in the car.
- Network $B2$ represents information about the user's state. It contains the evaluation of *Stress*, *Mood*, *Availability*, and in particular the static inference network for human *Motion Related Activity*, presented in section 5.3.3.2 on page 124. Moreover, three interface nodes to *Agenda and Time* and *High Level Activity* are contained. All this information stems from the user's local smart space and can be inferred in it.

[4] Statistics from `http://www.facebook.com/press/info.php?statistics`, as of 06/06/2011

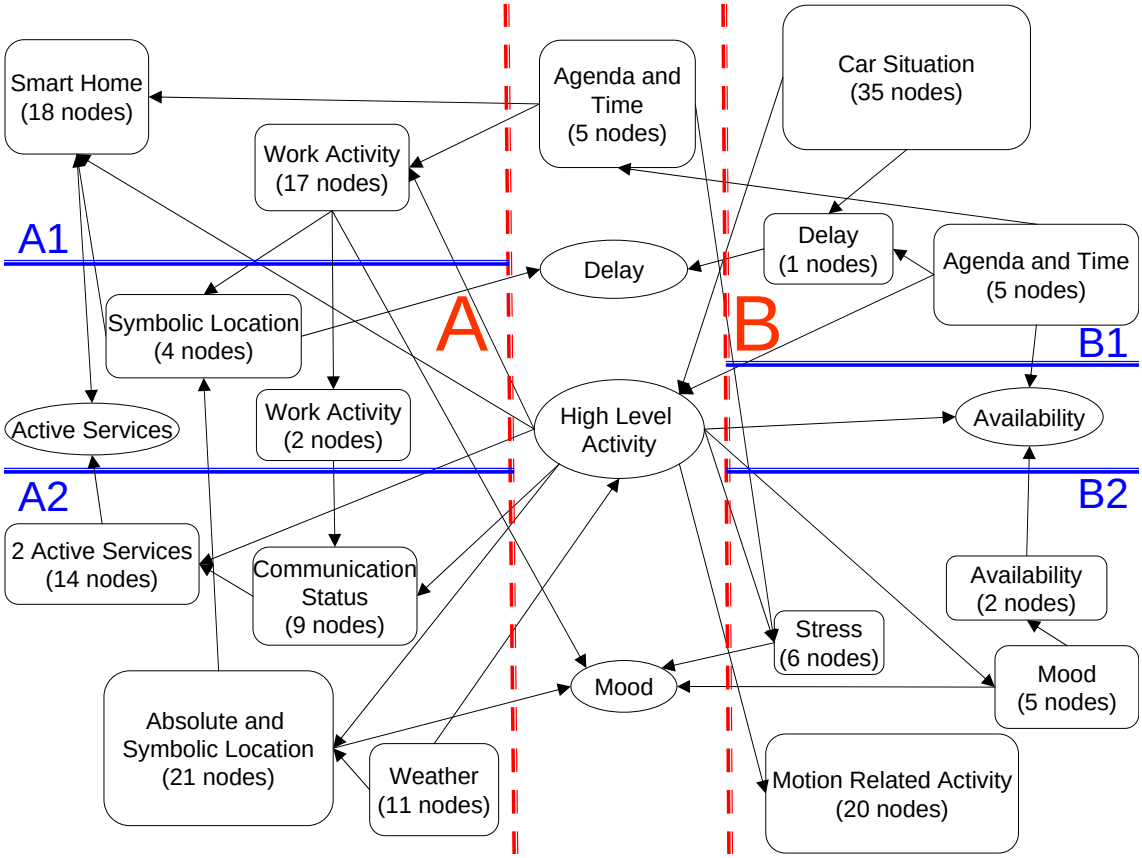


Figure 6.24: Modular split of the *Situation* BN from Figure 6.23 into four parts. BN A on the left side, containing *Work Activity*, *Location* and *Smart Home*, is connected to BN B on the right side (with *Motion Related Activity*, *Availability* and the *Car Situation*) only via eight overlapping nodes from *Agenda and Time*, the *High Level Activity*, *Mood* and *Delay*. Similarly, networks A and B are split into $A1$ and $A2$ (sharing seven nodes), and $B1$ and $B2$ (sharing only node *Availability*), respectively.

These four Bayesian networks are used to show the benefits of evaluating Bayesian networks separately in modules linked by soft evidence, as proposed in section 5.4, pp. 129 et seqq. Although more than 50 Bayeslets have been identified in the *Situation* BN, the larger size and limited number of the BNs $A1$, $A2$, $B1$, $B2$ better illustrates the benefits of the proposed approach.

The basic reference for inference duration in this evaluation is the probabilistic inference in the monolithic *Situation* BN using the author's Java implementation of the PPTC algorithm on a notebook with Intel Core 2 Duo T9600 processor at 2.8 GHz and 3 GB of RAM under Windows XP. The measured reference times, shown in Table 6.6, are based on 100 executions without any evidence, because the duration of propagating probabilities through the junction tree is independent from evidence in the BN.

Inference steps	mean (in s)	standard deviation (in s)
BN construction	0.34	0.1
J-Tree construction	4.40	0.1
J-Tree initialisation	193.7	3.4
J-Tree propagation	203.8	2.5

Table 6.6: Duration (in s) of the steps of probabilistic inference in the *Situation* BN from Figure 6.23.

The evaluation shows that the construction of the object representation of the BN is relatively efficient and also the set up of the junction tree does not take too long, compared to the large size with 180 nodes and 275 edges. What takes really long are all operations on the potentials of the clusters, as their size is the cross product of the value ranges of all contained nodes. Although also the initialisation of the potentials in the junction tree takes more than three minutes on average, the biggest problem is the duration of the propagation of the conditional probabilities between the clusters. This is the actual inference which is repeated for every inference request when evidence is available. Taking 3.4 minutes on a laptop computer, it is obvious that inference of such a BN would not be tractable for mobile devices.

Although the Java implementation is far from optimal in terms of computational efficiency, the results show clearly the expected behaviour, e.g. compared to the inference in the BN for *Work Activity*. As has been shown in section 6.3.3, propagation in the 17 nodes comprising BN takes less than 2.5 ms.

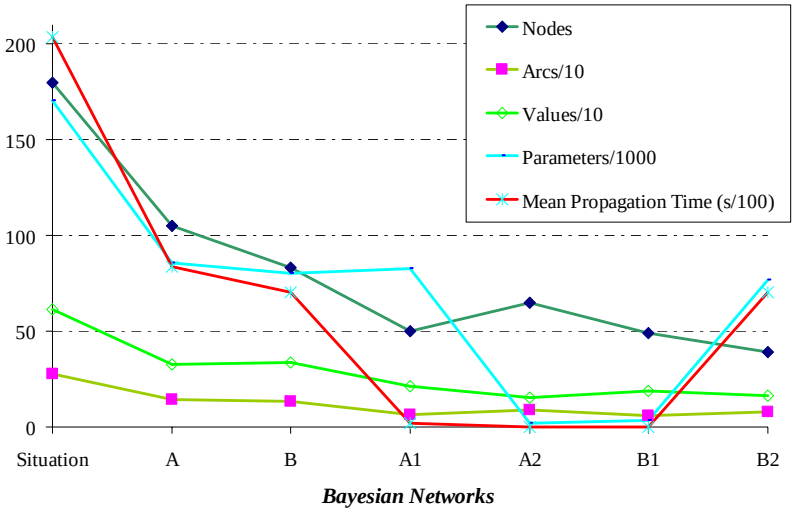
The propagation time of the sub-networks A, B, $A1$, $A2$, $B1$, and $B2$ in relation to their size is evaluated and compared to the *Situation* BN in Figure 6.25.

Graph (a) relates the absolute propagation times to the number of nodes, the number of arcs and the number of values – hence the basic factors targeted by the concepts of section 5.4. In addition, the number of parameters, i.e. the number of conditional probabilities necessary to fully specify the BN is given. To ease comparability of the curves for the different factors, the following scaling was used: the number of parameters has been divided by 1000, the number of arcs and values divided by 10 and compared to the propagation time in hundredths of seconds. The curves for the nodes, parameters and the mean propagation time develop similarly. Both, networks A and B, have roughly half of the nodes of the original BN, their individual inference time is under half of the full inference time. The drastic drop is experienced in the inference time for networks $A1$ (2.05), $A2$ (0.005) and $B1$ (0.12), although the number of nodes, again, is only divided in halves. The behaviour of $B2$ differs significantly. This network contains the Bayeslet for motion related activity, the only part of the *Situation* BN which has been learnt from data and not been specified manually. It has large value ranges and is strongly interconnected, which results in the high number of parameters, explaining the long inference time.

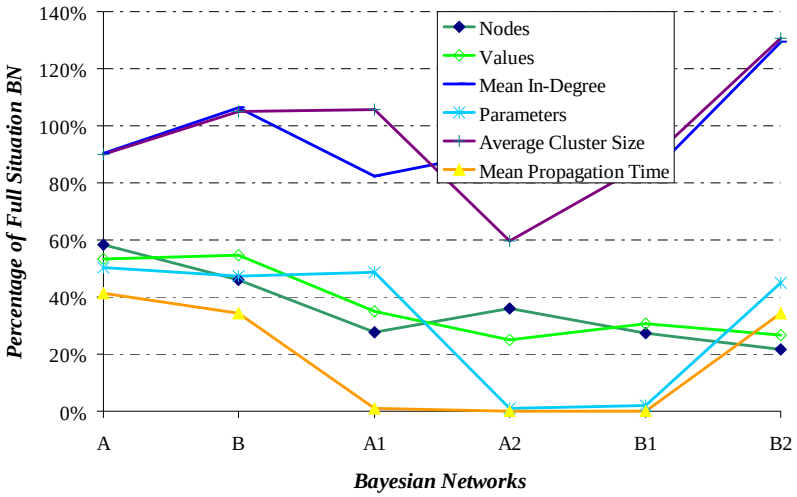
Graph (b) of Figure 6.25 details this issue further. In a relative scale according to the corresponding factors of the full *Situation* BN, it compares again nodes, values, parameters, propagation time and, newly, the mean number of incoming edges per node (*Mean In-Degree*) and the mean cluster size in the junction tree. The latter help explaining why network $A1$ has such a short inference time, although the number of parameters is quite as high as for networks A, B, and $B2$. While it is explainable for networks A and B by the number of nodes (A: 105, B : 83 and $A1$: 50), network $B2$ with 39 nodes is even smaller. $B2$ has however exceptionally high values for the mean in-degree and, consequently, the mean cluster size. The cluster size had been identified as a core factor for the computational complexity of arbitrarily structured BNs [85].

More concretely, the inference time savings are shown in Figure 6.26 for the full *Situation* BN and four combinations of the four sub-networks which can be used to infer a user's *Interaction Situation*, part of the group *Communication Status* (see Figure 6.23) in network $A2$. The following combinations are compared:

- $A + B$: inference of the full network in the two sub-networks A and B with soft evidence in the 8 interface nodes.



(a)



(b)

Figure 6.25: Comparison of the inference time (in terms of the propagation duration) of different Bayesian networks in relation to the factors determining the complexity of the Bayesian network. Absolute numbers in (a), and relative to the reference measures of the full *Situation* BN in (b).

- $A1 + A2 + B1 + B2$: inference of the full network in the four sub-networks $A1$, $A2$, $B1$ and $B2$ with soft evidence for 23 duplicated interface nodes.

- $A2 + (A1, B1, B2)$: the same nodes are used as for $A1 + A2 + B1 + B2$, however the spatial distribution of inference is used. $A2$ and $B2$ are inferred locally, $B1$ and $A1$ mainly remotely. Therefore, BN $A2$ with the queried node *Interaction Situation* can trigger evaluation of $A1$, $B1$ and $B2$ simultaneously. The maximum inference time of the other subnets added to the queried one gives the overall inference time.

- $A2 + (A1, B1)$: distributed inference for the non-queried sub-networks taking into account the dynamic plugging of inference networks proposed in section 5.4.4. Calculating the $NetEU$ of the available sub-networks, $B2$ has not been selected for composition due to its very high computational inference costs.

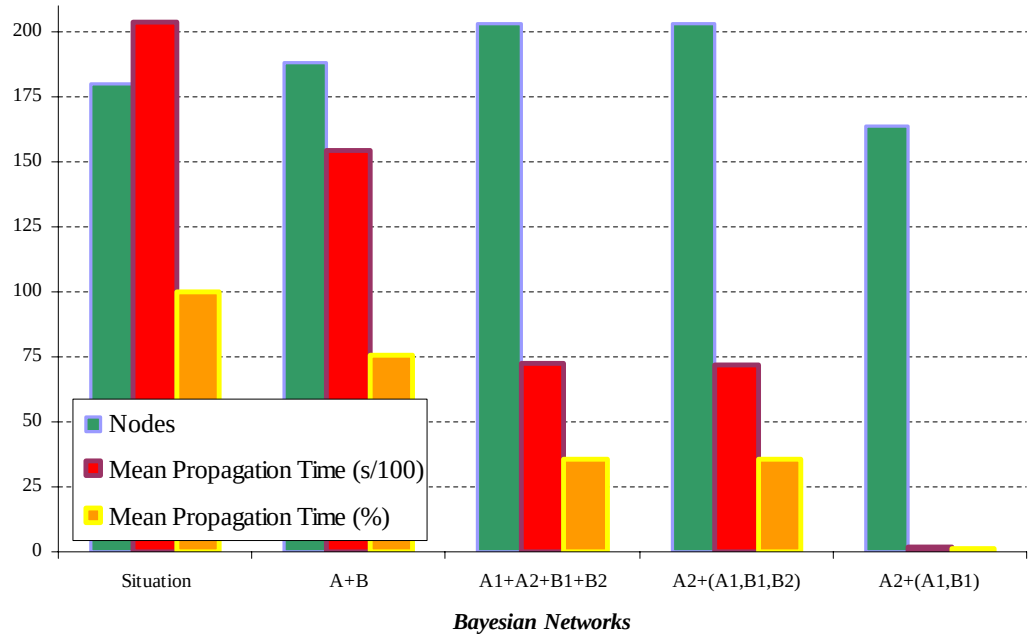


Figure 6.26: Comparison of the duration to infer a user's *Interaction Situation*, part of the group *Communication Status* in network $A2$, with different combinations of BNs.

Figure 6.26 shows for these evaluations that already $A+B$, hence dividing the *Situation* BN in halves, reduces inference time by 25 %. Inference partitioned into four sub-networks in $A1+A2+B1+B2$ decreases it to only a little more than a third of the original duration. The decrease from $A1 + A2 + B1 + B2$ to $A2 + (A1, B1, B2)$ is negligible, as inference in $A1$ and $B1$ which is parallelised with $B2$ only takes fractions of a second. Applying then the dynamic composition criteria in $A2 + (A1, B1)$ reduces inference time then to a mere percent (2.05 hundredths of a second) of the original inference time, while still processing 164 nodes, 91 % of the 180 original nodes.

Moreover, the more than 70 s propagation time in $B2$ are due to the basic, naïve approach. With the methods for the reductions of edges and value ranges, shown in sections 5.4.2 and 5.4.3 respectively, pp. 136 et seqq., inference time could be reduced significantly. Even better results are yielded by the approach used for the real-time activity recognition system evaluated in section 6.2: using the fact that input nodes can be defined in Bayeslets and therefore can be regarded as observed, more efficient inference algorithms can be applied. Like this, the 20 nodes with a *Mean In-Degree* of 2.65 in the Bayeslet shown in Figure 5.26 on page 124 is evaluated in 7.2 ms.

6.5.3 Remote Communication

Beyond the reduction of inference complexity, a second aim of the proposed methodology is to reduce the remote communication necessary to incorporate evidence from remote information sources. A first step to realise this is the proposed distributed inference approach in the user's smart space instead of a central approach. As a large part of the information concerning a user's situation is available in his smart space, transmission of this information is unnecessary.

The proposed methodology infers information in Bayeslets locally in the smart spaces where the information is generated, hence without remote communication demands. Re-

mote communication is necessary when results of a local inference are relevant for a remote Bayeslet and have to be transferred there to be introduced as soft evidence. Hence, instead of the transmission of all evidence used for the inference of a Bayeslet, only the outcome has to be transferred.

Therefore, the communication savings depend on the contents of the Bayeslets. A Bayeslet containing only one input node (like for instance the **RLD** Bayeslet for the calculation of the lateral distance based on a car's radar used in 6.4.1) could not save remote communication costs, unless it is used only locally. In contrast, the Bayeslet for *Motion Related Activity*, presented in Figure 5.26 and used in section 6.2 contains 19 input nodes, the always observed features of the IMU. Using this information remotely, consequently only one evidence would have to be transferred instead of 19.

A larger effect for the reduction of network traffic is achieved by the modular approach and the dynamic composition of Bayeslets. Avoiding the inclusion of a remote Bayeslet not only reduces inference time, but also saves network bandwidth as its soft evidence does not have to be retrieved. The modular approach in general allows that not all possibly related information has to be evaluated together – in terms of the example above hence, not all 130 facebook-friends have to be connected, not all meetings, all smart homes and cars have to be modelled with random variables in a BN. Being able to select only those Bayeslets of relevance for a task therefore inherently reduces the network traffic to all these remote domains to a fraction.

Moreover, the dynamic composition takes into account remote communication costs and can therefore help to limit it. Section 6.4.4 proposes to use a cost function which dynamically uses the current network load to rate the relevancy of a Bayeslet. If the channel is free, there is no need to impose particular limits, while in a full channel only the most important information should be transferred, i.e. those with the highest $NetEU$ taking into account computational, as well as communication costs.

6.5.4 Discussion

The example in this section has shown that the proposed methodology offers a set of concepts helping to improve the tractability of context inference on resource constrained devices.

It can be seen that the modular approach with soft evidence alone, as followed by the Bayeslet concept, already reduces the inference time in the example to 35 %. It can be even further reduced to only about 1 % by the application of a more suitable context inference method exploiting the assumptions defined by the Bayeslet concept, i.e. that there are input nodes which can be assumed to be always observed.

The same result would be obtained by the application of the proposed method for dynamic composition of Bayeslets based on the net expected utility $NetEU$ where the costs are evaluated against the benefits of incorporating new information.

The modular approach and the cost based composition of Bayesian inference rules moreover reduce the demands for remote communication, because only one transmission per Bayeslet is necessary and the number of connected Bayeslets can be reduced. This is easy to understand, but difficult to quantify, as a simulation environment would have to take into account the complete context of several users, their active services and context requests, different context information update frequencies, different background network load, etc.

The exact improvements for tractability, both for remote communication and computational demands, depend on the network size, structure and parameters of the individual Bayeslets. In the examples for location, human motion related activity and the work activity, it could be seen that networks with about 20 nodes are tractable with inference times of only a few milliseconds. Re-usability of Bayeslets however is improved when smaller entities are grouped, like the five Bayeslets shown in Figure 6.13. If they are too small however, e.g. Bayeslets for only one input and output node like in Figure 6.18, neither reduce remote communication (unless it is used in a Bayeslet in the same smart space only), nor inference time, as the node's outcome could be directly inserted into the connected node as soft evidence.

These considerations have to be taken into account by the creators of Bayeslets and could also be used to configure automatic Bayeslet learning processes.

Having seen the reductions possible by the proposed set of methods, the requirements of mobile context inference seem to be realistically achievable. It will be possible to infer the seven continuously monitored high level context attributes per person assumed in Table 3.1 (p.55) on mobile devices with the modular approach, dynamic reduction of value ranges, and intelligent composition decisions. Current limitations will be even more reduced by an optimised implementation of the inference algorithms and the further development of mobile devices.

Chapter 7

Summary and Conclusions

Concluding this work, this chapter summarises the work presented in the previous chapters and stresses the author's main contributions to the field of research. After that, the conclusions drawn from the evaluations are described and an outlook for future work in this field is given.

7.1 Summary

The research in this thesis has proposed to use Bayesian techniques (presented in chapter 2) for inference of context information. Central, server based approaches to context inference would not scale for the assumed high dimensions for context usage in ubiquitous computing systems, as can be understood from the analysis in chapter 3. Distributed context inference in every user's smart space however suffers from the resource constraints of the mobile devices which often are the core part of a personal smart space. These limitations are severe, as Bayesian inference is NP-hard and no general inference algorithm with less than exponential complexity is known.

To allow for efficient response times to context requests in ubiquitous computing, this research has proposed in section 5.1 an integration of context inference in context management systems which allows for both flexible creation of inference rules and resource efficient inference scheduling based on actual demands. To allow for distributed inference, reduced inference rule size, personalisation of inference rules and better privacy control, Bayesian networks are divided in modules, so called Bayeslets.

Chapter 5 has shown two examples for Bayeslets with particular importance for context – location (in section 5.2) and motion related activity (in section 5.3). An efficient approach for the fusion of a Wi-Fi fingerprinting with an inertial navigation system for precise location inference with a Kalman filter has been presented in section 5.2, as well as the usage of this information (i) to extract symbolic location maintaining the important uncertainty information and (ii) to deduce the proximity of other entities measuring the distance based on a known map with a gas diffusion model. Section 5.3 has shown a fully Bayesian approach to recognise human motion related activity. It has described the features computed from the input data with their justification, the automated learning of a static Bayesian network and four different classifiers which can be used to infer the activity.

Section 5.4 has detailed the Bayeslet concept, showing how all factors determining the complexity of probabilistic inference (nodes, edges and value ranges) can be addressed by approaches to make context inference better tractable. The current situation of a user thereby is incorporated in the decisions how to reduce the value ranges and which

Bayeslets to combine for a given inference target. The outcome of a Bayeslet is used as soft evidence in a connected Bayeslet.

The application of the proposed concepts and methodologies has been shown in example environments in chapter 6. The location estimation approach yields an average position error of about 1.6 m, the additional time necessary to extract the symbolic location only amounts to about 1 %. Human motion related activity recognition yields its best results when inferred with a grid based filter based on a learnt Bayesian network structure, reaching recall rates between 93 and 100 %. The dynamic reduction of value ranges has yielded inference time savings between 75 and 90 %. The increased effort for constructing and maintaining the dynamic value ranges will have amortised at the fourth inference in a reduced Bayeslet at latest. The best results thereby are reached with a *protection extension (PE)* approach which takes into account the context values which are of interest for the requester and excludes them (and closely related other values) from reduction. The application of the criteria for the composition of Bayeslets has shown that both mutual information and value of information are suitable measures that complement each other, giving the Bayeslet designer the choice to select the more appropriate for the specific Bayeslet.

Applying the different proposed concepts (modularisation, composition of only relevant Bayeslets and assumptions about the interface nodes) to an example in section 6.5 has yielded strong reductions in inference time, up to 99 % and has suggested also reductions in remote communications.

7.2 Main Contributions

The main contributions of this work are the following:

1. A model for generic context inference:
   This work has proposed a general method for the modelling and usage of context information in context inference. Bayesian inference rules, called Bayeslets, have to represent closed domains which can be dynamically selected and plugged for a given inference target using the novel approach. The resulting Bayesian inference modules have the advantages of object oriented modules and are reusable for different domains. This concept is generically applicable and has been demonstrated in different domains, such as road traffic, working life or the physical status of people, for raw data and high level information.

2. Faster Bayesian Context Inference:
   As the computational complexity of Bayesian inference is NP-hard, context inference has to minimise the factors that determine computation time. These are the number of random variables in the BN, the number of edges between them and the number of values in the value ranges of the random variables. This work has proposed countermeasures for all of them and demonstrated their benefits in different application examples. Using Bayeslets, the number of random variables is reduced, as well as the number of edges. Edges to external Bayeslets are only added where they are relevant. Learning personalised Bayeslets from data also reduces the dependencies and therefore the edges to those necessary for a given user. The number of values per value range finally is addressed by the concept of dynamic value ranges, depending on the user's current context. Different examples have shown that inference in Bayeslets with a size of approximately 20 nodes is tractable for mobile devices, while the difference to the ground truth (or exact inference) is acceptable. The additional

assumptions on the structure of Bayeslets, always observed input nodes and a limited number of output nodes, have proven useful for the selection of appropriate inference techniques and to allow for selective composition of Bayeslets.

3. Location Inference:
   For context inference a translation of absolute location to symbolic location is necessary, where the uncertainty information has to be maintained which is most important for context inference. This work has proposed and demonstrated a viable approach based on a particle filter, as well as a formalism for modelling several random variables describing location aspects in a single Bayeslet. This formalism is necessary to avoid inconsistent random variables (containing not mutually exclusive values) and to allow for generic usability of symbolic location in Bayesian context inference. Also a proposal for inference of proximity based on precise absolute location information, existing floor plans and a gas diffusion model has been shown.

4. Highly reliable, unobtrusive motion related activity recognition:
   This work has provided an inference module for the activities *sitting, standing, walking, running, jumping, falling* and *lying* based on a single, belt-worn sensor box. A fully Bayesian approach has been used to learn a Bayesian network structure together with the corresponding conditional probabilities and to infer the current activity. The resulting classifier has proven robust to different application environments, different motion styles and different users, as well as being computationally so efficient that it would not be a major challenge to integrate the complete recognition module in a belt buckle.

The concepts proposed and demonstrated in this thesis help to answer the research questions presented in section 1.3 on page 11. The core idea to realise the integration of inference rules into the context management system, to adapt and personalise them during runtime, and to allow for compatibility with other inference methods is the modular approach of the Bayeslet concept with well defined interfaces. Only the reduced size of Bayesian networks makes them realistically manageable. The integration in context management systems and with other inference methods is described in section 5.1, answering research questions (4), (5) and (8).

Personalised context inference, research question (3), is a consequence of minimising the Bayeslets strictly to the necessities. The BN Structure learning and further processes described in section 5.4.2 yield this result. The adaptation to the changing mobile environments, research question (2), is solved by the modular approach and the situation aware composition of modules proposed in section 5.4.4 and demonstrated in section 6.4. The composition criteria measure the current relevancy of a Bayeslet in the current situation, addressing research questions (6) and (7).

The examples presented in this thesis show that the Bayesian approach to context inference is applicable to both, low and high level context, cf. research question (9). The approaches for lower level context, location and human motion related activity, have been presented and evaluated in detail in the sections 5.2 and 6.1, 5.3 and 6.2, respectively. An example for high level context inference is given in the evaluation of the professional activity at a user's workplace in section 6.3.1. The applicability hence does not depend on the level of information, but only on careful modelling of the influence factors and the available information. The level of context rather has an influence on the selection of the Bayesian approach to be used, hence the decision whether the involved random variables are continuous or discrete, if a transition model has relevancy or not.

The answer to research questions (1) and (10) is given by the overall set of methods proposed. The advantages are the general applicability to all domains, efficient inference and reduced communication needs, enabled by the orchestration of the concepts in section 5.4. The improvements have been shown for an example in section 6.5.

7.3 Conclusions

The amount of information digitally available constantly increases, as new smartphones and other electronic devices emerge and spread. This information allows for informed inference of higher level context, but also poses the problem of inference complexity. Although also the hardware constantly improves (approximately by Moore's law [7]), it cannot keep pace with the increasing requirements of context inference, because for both, logical and probabilistic inference, only algorithms with exponential complexity are known.

Therefore this research has proposed methods to limit the amount of information considered for context inference, hence to trade exactness of inference against efficiency gain. From the application examples shown in this work, it can be concluded that the proposed way is viable and that not a single concept, but a combination of approaches is necessary to minimise resource consumption of inference.

With the general applicability of the proposed concepts to multiple domains, this thesis combines a number of research directions which have been rather decoupled, like context modelling, activity recognition, multi-sensor location estimation and machine learning for proactive context awareness.

7.4 Future Work

Many new insights have been obtained in this research. In some of the examined application areas however, the new ideas only show the potential without providing a mature, market-ready solution. The following areas for future work have been identified:

1. User acceptance analysis:
   The most crucial issue for the success of ubiquitous, context aware computing is the users' acceptance. It depends on trustworthiness of the system providers, privacy protection, usability of devices, but also on context inference. Inference quality obviously has to be high enough to avoid frustration which would arise when the system misinterprets the situation too often and then always causes wrong actions. But also the acceptable response time is a crucial parameter for context inference. If a limit for an acceptable response time could be identified (e.g. per application), real-time inference approaches could be employed which allow for the connection of more Bayeslets and therefore improve inference quality as far as possible within the specified time frame.

2. Cost functions:
   Taking care of a maximum response time could be realised for instance with an adaptive cost function for the composition of Bayeslets. In general, the work on cost functions for composition still offers room for research. A range of requirements (speed of computation, adaptation to situation, comparability among different Bayeslets, etc.) can be defined which has to be fulfilled for all different domains causing composition costs, such as time, memory and communication costs.

3. Further user context:
   User acceptance also depends on context awareness, as a high number of sensors

which the user would have to wear constantly would reduce the usability. The current work already uses sensors in the shoe and worn at the belt. Further research with microphones and sensors in a chest belt measuring the wearer's psychophysiology is currently being undertaken. On the one hand, the modular approach proposed in this work eases the integration of new modules which determine for instance the user's stress level, on the other hand the usability of the system has to be considered. At a first step probably, only users with special needs (e.g. continuous health monitoring in AAL) would accept obtrusion by a ubiquitous system when for them the benefits outweigh the inconveniences.

4. Interaction recognition:
   An important factor for the recognition of a user's situation are his interactions with other people. This information is useful for the composition of Bayeslets themselves, but also for context aware services, advertising, social networking, etc. Important input for the recognition of interactions are location, orientation, sound, but also active remote communication channels. Inference could be based on similar techniques like the recognition of motion related activity.

5. Group context:
   Further current research, as in the EU research project ICT Societies[1] combines personal, context aware smart spaces with the benefits of social computing. To this end, not only the context of single users, but also of groups of persons has to be inferred. Also this inference requires a modular approach and dedicated modelling of a group's common interests and objectives. A group utility function could represent such characteristics and extend a Bayeslet for inference to take decisions for proactive service execution.

Further work in these areas will lead to both, better context inference results and better usability. This shall increase the user acceptance and eventually make the vision of ubiquitous, context aware computing come true.

[1] http://www.ict-societies.eu/

Chapter 8

References

8.1 Ubiquitous Computing and Context Awareness

[1] G. D. Abowd and B. N. Schilit. Ubiquitous computing: the impact on future interaction paradigms and hci research. In *Proceedings of CHI '97. Extended abstracts on Human factors in computing systems: looking to the future*, CHI EA '97, pages 221–222, New York, NY, USA, 1997. ACM.

[2] ALPS. Products in everyday life. Body control power train. Online: `http://www.alps.com/e/about_alps/mar/mar_body.html`, 2011. Retrieved: March 1, 2011.

[3] Y. Aoyagi and R. J. Shephard. Steps per day: The road to senior health? *Sports Medicine*, 39(6):423–438, 2009. Adis Online, ISSN: 0112-1642.

[4] Apple. Press release. Apple's app store downloads top 10 billion. Online: `http://www.apple.com/pr/library/2011/01/22appstore.html`, Jan. 2011.

[5] I. Bagrak. Toward a structured approach to wireless sensor network design. EE Times, Oct. 2008.

[6] T. Becks, B. Eberhardt, S. Heusinger, S. Pongratz, and J. Stein. Intelligente Heimvernetzung. Komfort – Sicherheit – Energieeffizienz – Selbstbestimmung. *VDE-Positionspapier*, Jan. 2010. Online: `http://www.vde.com/de/Institut/Querschnittstechnologien/IntelligenteHeimvernetzung/Documents/IS%20604D.pdf`.

[7] D. C. Brock, editor. *Understanding Moore's Law: Four Decades of Innovation.* Chemical Heritage Press, Sept. 2006. ISBN: 978-0941901413.

[8] T. Buchholz and M. Schiffers. Quality of context: What it is and why we need it. In *Proceedings of the 10th Workshop of the OpenView University Association: OVUA03*, 2003.

[9] A. K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001. Springer-Verlag, London, UK, ISSN: 1617-4909.

[10] A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2):97–166, December 2001. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, ISSN: 0737-0024.

[11] K. Dolinar, E. Papadopoulou, N. Liampotis, and Y. Abu Shaaban. Protecting the privacy of personal smart spaces. In S. M. McBurney and E. Papadopoulou, editors, *PERSIST Workshop on Intelligent Pervasive Environments*, Proceedings of the Symposium PERSIST Workshop on Intelligent Pervasive Environments, pages 33–40. SSAISB, 2009.

[12] M. R. Ebling, G. Hunt, and H. Lei. Issues for context services for pervasive computing. In *Proceedings of the Advanced Workshop on Middleware for Mobile Computing*, Heidelberg, Germany, 2001. Springer.

[13] European Commision. Comparable time use statistics. national tables from 10 european countries. *eurostat. Methodologies and Working papers*, pages 1–14, Feb. 2005.

[14] European Commision. Harmonised european time use surveys. 2008 guidelines. *eurostat. Methodologies and Working papers*, pages 1–206, 2008.

[15] W. Fitzgerald, K. Doolin, F. Mahon, C. Hauser, A. F. Gomez-Skarmeta, S. Butler, P. Schlosser, and B. Weyl. Daidalos security framework for mobile services. In *Proceedings of eChallenges 2005*, June 2005.

[16] R. L. Forge. Clinical use of pedometers. brief overview and instructions for health care providers. Online: `http://www.lipid.org/education/courses/ccrr/assets/STEPcounterInstructions10HCP.pdf`, 2010.

[17] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199 – 220, 1993. Academic Press Ltd. London, UK, ISSN: 1042-8143.

[18] T. Gu, X. H. Wang, H. K. Pung, and D. Zhang. An ontology-based context model in intelligent environments. In *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference*, pages 270–275, San Diego, CA, USA, Jan. 2004. Society for Computer Simulation.

[19] K. Henricksen, J. Indulska, and A. Rakotonirainy. Modeling context information in pervasive computing systems. In F. Mattern and M. Naghshineh, editors, *Pervasive '02: Proceedings of the First International Conference on Pervasive Computing*, pages 79–117, London, UK, 2002. Springer-Verlag.

[20] R. C. Hilborn. Sea gulls, butterflies, and grasshoppers: A brief history of the butterfly effect in nonlinear dynamics. *American Journal of Physics*, 72(4):425–427, 2004. American Association of Physics Teachers, ISSN: 0002-9505.

[21] International Telecommunication Union. ITU corporate annual report 2009. Online: `http://www.itu.int/dms_pub/itu-s/opb/conf/S-CONF-AREP-2009-PDF-E.pdf`, 2009.

[22] International Telecommunication Union. Global ICT developments, 2000-2010. Online: `http://www.itu.int/ITU-D/ict/statistics/material/excel/2010/Global_ICT_Dev_00-10.xls`, Oct. 2010.

[23] International Telecommunication Union. Mobile broadband subscriptions per 100 inhabitants, 2000-2010. Online: `http://www.itu.int/ITU-D/ict/statistics/material/excel/2010/Mobile_bb_00-10.xls`, Oct. 2010.

[24] International Telecommunication Union. The world in 2010: ICT facts and figures. Online: `http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf`, Oct. 2010.

[25] S. L. Kiani, M. Knappmeyer, N. Baker, and B. Moltchanov. A federated broker architecture for large scale context dissemination. In *Proceedings of International Conference on Computer and Information Technology*, pages 2964–2969, Los Alamitos, CA, USA, 2010. IEEE Computer Society Press.

[26] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, and M. Spasojevic. People, places, things: Web presence for the real world. *Mobile Networks and Applications*, 7(5):365–376, Oct. 2002. Springer Netherlands, ISSN: 1383-469X.

[27] M. Krauss and M. Bogen. Conveying cultural heritage and legacy with innovative ar-based solutions. In J. Trant and D. Bearman, editors, *Museums and the Web 2010: Proceedings*. Archives & Museum Informatics, Mar. 2010.

[28] National Research Council Canada. Wireless sensors for smart buildings. Online: `http://www.nrc-cnrc.gc.ca/eng/news/nrc/2009/02/09/wireless-sensors.html`, Feb. 2009.

[29] K. Nice. How car computers work. Online: `http://auto.howstuffworks.com/under-the-hood/trends-innovations/car-computer.htm/printable`. Retrieved: February 25, 2011.

[30] B. O'Conaill and D. Frohlich. Timespace in the workplace: dealing with interruptions. In *Conference companion on human factors in computing systems*, CHI '95, pages 262–263, New York, NY, USA, 1995. ACM.

[31] K. Pattison. Worker, interrupted: The cost of task switching. Fast Company. Online: `http://www.fastcompany.com/articles/2008/07/interview-gloria-mark.html`, July 2008. Retrieved: 2/3/2011.

[32] T. Pfeifer and S. van der Meer. The active store providing quality enhanced unified messaging. In *Proceedings of the 5th Conference on Computer Communications, AFRICOM-CCDC'98*, pages 279–293, Oct. 1998.

[33] S. Pietschmann, A. Mitschick, R. Winkler, and K. Meiï¿œner. Croco: Ontology-based, cross-application context management. In *Proceedings of Third International Workshop on Semantic Media Adaptation and Personalization*, pages 88–93, Los Alamitos, CA, USA, Dec. 2008. IEEE Computer Society Press.

[34] C. Pils, I. Roussaki, T. Pfeifer, N. Liampotis, and N. Kalatzis. Federation and sharing in the context marketplace. In J. Hightower, B. Schiele, and T. Strang, editors, *Location- and Context-Awareness. Proceedings of Third International Symposium, LoCA 2007*, volume 4718 of *Lecture Notes in Computer Science*, pages 121–138. Springer Berlin / Heidelberg, 2007.

[35] O. Riva. Contory: a middleware for the provisioning of context information on smart phones. In *Middleware '06: Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware*, pages 219–239, New York, NY, USA, 2006. Springer-Verlag New York, Inc.

[36] O. Riva and S. Toivonen. The DYNAMOS approach to support context-aware service provisioning in mobile environments. *Journal of Systems and Software*, 80(12):1956–1972, 2007. Elsevier Science Inc., ISSN: 0164-1212.

[37] M. Satyanarayanan. Pervasive computing: vision and challenges. *Personal Communications, IEEE*, 8(4):10–17, Aug. 2001. IEEE Communications Society, Washington, DC, USA, ISSN: 1070-9916.

[38] J. M. Serrano, J. Serrat, and A. Galis. Ontology-based context information modelling for managing pervasive applications. In *ICAS '06: Proceedings of the International Conference on Autonomic and Autonomous Systems*, pages 47–52, Washington, DC, USA, 2006. IEEE Computer Society Press.

[39] T. Strang. Service-Interoperabilität in Ubiquitous Computing Umgebungen. PhD thesis, LMU München, 2004.

[40] T. Strang and C. Linnhoff-Popien. Service interoperability on context level in ubiquitous computing environments. In *Proceedings of International Conference on Advances in Infrastructure for Electronic Business, Education, Science, Medicine, and Mobile Technologies on the Internet (SSGRR2003w), L'Aquila/Italy, January 2003. [Online]. Available: citeseer.nj.nec.com/st*, 2003.

[41] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Proceedings of Workshop on Advanced Context Modelling, Reasoning and Management in cooperation with UbiComp 2004: The Sixth International Conference on Ubiquitous Computing*, Nottingham, UK, Sept. 2004.

[42] T. Strang, C. Linnhoff-Popien, and K. Frank. Cool: A context ontology language to enable contextual interoperability. In J.-B. Stefani et al., editor, *Proceedings of International Conference on Distributed Applications and Interoperable Systems (DAIS 2003)*, volume 2893 of *Lecture Notes in Computer Science*, pages 236–247. Springer-Verlag, Heidelberg, November 2003.

[43] V. Suraci, S. Mignanti, and A. Aiuto. Context-aware semantic service discovery. In *Proceedings of 16th IST Mobile and Wireless Communications Summit*, pages 1–5, Los Alamitos, CA, USA, July 2007. IEEE Computer Society Press.

[44] United Nations. Department of Economic and Social Affairs. Population Division. World population to 2300. Online: `http://www.un.org/esa/population/publications/longrange2/WorldPop2300final.pdf`, 2004.

[45] M. Uschold, M. Gruninger, M. Uschold, and M. Gruninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review (Online)*, 11:93–136, 1996. Cambridge University Press, ISSN: 0269-8889.

[46] B. van Arem, C. J. van Driel, and R. Visser. The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics. *IEEE Transactions on Intelligent Transportation Systems*, 7(4), 2006. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 1524-9050.

[47] M. Weiser. The computer for the 21st century. In R. M. Baecker, J. Grudin, W. A. S. Buxton, and S. Greenberg, editors, *Human-computer interaction*, pages 933–940. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995. ISBN: 1-55860-246-1.

8.2 Bayesian Network Theory

[48] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, New York, NY, USA, 9th Dover printing, 10th GPO printing edition, 1964. ISBN: 978-0486612720.

[49] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 1053-587X.

[50] J. Bilmes. On virtual evidence and soft evidence in bayesian networks. Technical Report UWEETR-2004-0016, University of Washington, Dept. of Electrical Engineering, 2004. Online: `https://www.ee.washington.edu/techsite/papers/refer/UWEETR-2004-0016.html`.

[51] A. Bogomolny. What is probability? Online: `http://www.cut-the-knot.org/Probability/Dictionary.shtml`, 2011. Retrieved 20 March 2011.

[52] D. M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, March 2002. Microtome Publishing, ISSN: 1532-4435.

[53] I. D. Coope and C. J. Price. On the convergence of grid-based methods for unconstrained optimization. *SIAM Journal on Optimization*, 11(4):859–869, 2000. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, ISSN: 1052-6234.

[54] R. Dodier. A revision of the Bayesian network interchange format. Technical Report JCEM-TR-1998-9, University of Colorado, Mar. 1998. Online: `http://civil.colorado.edu/~dodier/publications.html`.

[55] F. Gagliardi Cozman. XMLBIF: version 0.3. Online: `http://www.cs.cmu.edu/javabayes/Home/index.html`, 2001.

[56] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Information Science and Statistics. Springer, July 2001. ISBN: 0-387-95259-4.

[57] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME. Journal of Basic Engineering*, Series D(82):35–45, 1960. American Society of Mechanical Engineers, ISSN: 0021-9223.

[58] M. Khider. Implementation of a simulator/demonstrator for the softlocation concept using Bayesian filters. Master's thesis, Universität Ulm, November 2005.

[59] F. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47:498–519, 2001. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 0018-9448.

[60] P. Maybeck. *Stochastic Models, Estimation and Control, Volume I*, volume 141 of *Mathematics in Science and Engineering*. Academic Press, Inc., New York, 1979. ISBN: 0-12-480701-1.

[61] K. P. Murphy. *Dynamic Bayesian networks : representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

[62] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, 1988. ISBN: 978-1558604797.

[63] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000. ISBN: 978-0521895606.

[64] M. Pradhan, G. M. Provan, B. Middleton, and M. Henrion. Knowledge engineering for large belief networks. In R. L. de Mántaras and D. Poole, editors, *UAI '94: Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence, July 29-31, 1994, Seattle, Washington, USA*, pages 484–490. Morgan Kaufmann, 1994.

[65] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb. 1989. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 0018-9219.

[66] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995. ISBN: 978-0131038059.

[67] R. D. Shachter. Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In *Proceedings of the Fourteenth Conference in Uncertainty in Artificial Intelligence*, pages 480–487, 1998.

[68] A. N. Shiryayev, editor. *Selected Works of A.N. Kolmogorov: vol. 2 Probability Theory and Mathematical Statistics*. Springer, 1st edition, Feb. 1992. ISBN: 902772797X.

[69] T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, UAI '90, pages 255–270, New York, NY, USA, 1991. Elsevier Science Inc.

8.3 Learning Techniques

[70] I. Cohen, A. Bronstein, and F. G. Cozman. Online learning of Bayesian network parameters. Technical report, HP Laboratories Palo Alto. Internet Systems and Storage Laboratory, June 2001. HPL-2001-55R1. Online: `http://www.hpl.hp.com/techreports/2001/HPL-2001-55R1.pdf`.

[71] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 09(4):309–347, Oct. 1992. Springer Verlag, ISSN: 0885-6125.

[72] G. Elidan and N. Friedman. Learning hidden variable networks: The information bottleneck approach. *Journal of Machine Learning Research*, 6:81–127, Dec. 2005. Microtome Publishing, ISSN: 1532-4435.

[73] G. Elidan, N. Lotner, N. Friedman, and D. Koller. Discovering hidden variables: A structure-based approach. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Neural Information Processing Systems*, pages 479–485. MIT Press, 2001.

[74] N. Friedman. The Bayesian structural EM algorithm. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 129–138. Morgan Kaufmann, 1998.

[75] D. Heckerman. A Tutorial on Learning with Bayesian Networks. Technical Report MSR-TR-95-06, Microsoft Research, Redmond, Washington, 1995. Revised June 1996.

[76] D. Heckerman. A tutorial on learning with Bayesian networks. In D. Holmes and L. Jain, editors, *Innovations in Bayesian Networks*, volume 156 of *Studies in Computational Intelligence*, pages 33–82. Springer Berlin / Heidelberg, 2008. 10.1007/978-3-540-85066-3_3.

[77] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, and A. Wu. An efficient k-means clustering algorithm: analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):881–892, July 2002. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 0162-8828.

[78] A. Krause, A. Smailagic, and D. P. Siewiorek. Context-aware mobile computing: Learning context-dependent personal preferences from a wearable sensor array. *IEEE Transactions on Mobile Computing*, 5(2):113–127, 2006. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 1536-1233.

[79] M. Singh. Learning Bayesian networks from incomplete data. In *Proceedings of the fourteenth national conference on artificial intelligence and ninth conference on Innovative applications of artificial intelligence*, AAAI'97/IAAI'97, pages 534–539. AAAI Press, 1997.

8.4 Probabilistic Inference

[80] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic and Discrete Methods*, 8(2):277–284, April 1987. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, ISSN: 0196-5212.

[81] S. Cook. The complexity of theorem proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

[82] G. F. Cooper. Probabilistic inference using belief networks is NP-hard. Technical Report KSL-87-27, Medical Computer Science Group, Knowledge Systems Laboratory, Stanford University, Stanford, CA, 1987.

[83] P. Dagum and M. Luby. Approximating probabilistic inference in bayesian belief networks is NP-hard. *Artificial Intelligence*, 60(1):141–153, 1993. Elsevier Science Publishers Ltd., Essex, UK, ISSN: 0004-3702.

[84] A. Garvey and V. Lesser. A survey of research in deliberative real-time artificial intelligence. *Real-Time Systems*, 6:317–347, 1994. Springer Netherlands, ISSN: 0922-6443.

[85] H. Guo and W. Hsu. A survey of algorithms for real-time Bayesian network inference. In H. Guo, E. Horvitz, W. H. Hsu, and E. Santos, editors, *Working Notes of the Joint Workshop (WS-18) on Real-Time Decision Support and Diagnosis, AAAI/UAI/KDD-2002*. AAAI Press, July 2002.

[86] C. Huang and A. Darwiche. Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning*, 15(3):225–263, Oct. 1996. Elsevier Science Publishers Ltd., ISSN: 0888-613X.

[87] M. A. Kłopotek. Reasoning and learning in extended structured Bayesian networks. *Fundamenta Informaticae*, 58:105–137, April 2003. IOS Press, Amsterdam, The Netherlands, ISSN: 0169-2968.

[88] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):157–224, 1988. Blackwell Publishing for the Royal Statistical Society, ISSN: 0035-9246.

[89] J. Pearl. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, 29(3):241–288, 1986. Elsevier Science Publishers Ltd., Essex, UK, ISSN: 0004-3702.

[90] D. M. Pennock. Logarithmic time parallel Bayesian inference. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 431–438. Morgan Kaufmann, 1998.

[91] R. D. Shachter. Intelligent probabilistic inference. In *Proceedings of the Uncertainty in Artificial Intelligence Annual Conference on Uncertainty in Artificial Intelligence (UAI'85)*, pages 371–382, Amsterdam, NL, 1985. Elsevier Science.

[92] R. D. Shachter. Probabilistic inference and influence diagrams. *Operations Research*, 36:589–604, July 1988. Institute for Operations Research and the Management Sciences (INFORMS), Linthicum, Maryland, USA, ISSN: 0030-364X.

[93] D. Slezak, G. Wang, M. S. Szczuka, I. Düntsch, and Y. Yao, editors. *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, 10th International Conference, RSFDGrC 2005, Regina, Canada, August 31 - September 3, 2005, Proceedings, Part I*, volume 3641 of *Lecture Notes in Computer Science*. Springer, 2005.

[94] L. Smail. D-separation and computation of probability distributions in Bayesian networks. *Artificial Intelligence Review*, 31(1):87–99, 2009. Springer Netherlands, ISSN: 0269-2821.

[95] M. P. Wellman and C.-L. Liu. State-space abstraction for anytime evaluation of probabilistic networks. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 567–574. Morgan Kaufmann, 1994.

[96] D. Wu and C. J. Butz. On the complexity of probabilistic inference in singly connected Bayesian networks. In Slezak et al. [93], pages 581–590.

8.5 Information and Utility Theory

[97] E. J. Clarke and B. A. Barton. Entropy and MDL discretization of continuous variables for Bayesian belief networks. *International Journal of Intelligent Systems*, 15(1):61–92, Jan. 2000. John Wiley & Sons, Inc., New York, NY, USA, ISSN: 1098-111X.

[98] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *Proceedings of the International Conference on Machine Learning*, pages 194–202, New York, NY, USA, 1995. Morgan Kaufmann Publishers, Inc.

[99] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, volume 2, pages 1022–1027, New York, NY, USA, 1993. Morgan Kaufmann Publishers, Inc.

[100] A. V. Kozlov and D. Koller. Nonuniform dynamic discretization in hybrid networks. In D. Geiger and P. P. Shenoy, editors, *UAI '97: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 314–325, New York, NY, USA, Aug. 1997. Morgan Kaufmann Publishers, Inc.

[101] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003. ISBN: 0-521-64298-1.

[102] M. Röckl. *Cooperative Situation Awareness in Transportation*. PhD thesis, University of Innsbruck (Austria), 2010.

[103] M. Röckl, J. Gacnik, and J. Schomerus. Integration of Car-2-Car Communication as a Virtual Sensor in Automotive Sensor Fusion for Advanced Driver Assistance Systems. In A. of German Engineers (VDI), editor, *Proceedings of the FISITA 2008 World Automotive Congress*, pages 1–9, Munich, Germany, Sept. 2008. Springer Automotive Media.

[104] L. Savage. *The Foundations of Statistics*. Courier Dover Publications, 2nd edition, 1972. ISBN: 978-0486623498.

[105] C. E. Shannon and W. Warren. A Mathematical Theory of Communication. *Bell system technical journal*, 27(3):379–423, July 1948. University of Illinois Press, ISSN: 0005-8580.

8.6 Context Inference

[106] A. Aamodt and E. Plaza. Case-based reasoning; foundational issues, methodological variations, and system approaches. *AI Communications*, 7:39–59, March 1994. IOS Press, Amsterdam, The Netherlands, ISSN: 0921-7126.

[107] M. Angermann, P. Robertson, and T. Strang. Issues and requirements for bayesian approaches in context aware systems. In Strang and Linnhoff-Popien [136], pages 235–243.

[108] B. Beamon. Evaluation of first order Bayesian networks for context modeling and reasoning. In *Proceedings of Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pages 867–868, Mar. 2010.

[109] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161–180, April 2010. Elsevier Science Publishers B.V., Amsterdam, The Netherlands, ISSN: 1574-1192.

[110] A. Bikakis, T. Patkos, G. Antoniou, and D. Plexousakis. A survey of semantics-based approaches for context reasoning in ambient intelligence. In M. Mühlhäuser, A. Ferscha, and E. Aitenbichler, editors, *Constructing Ambient Intelligence, Proceedings of AmI-07 Workshops*, volume 11 of *Communications in Computer and Information Science*, pages 14–23. Springer Verlag, 2008.

[111] S. Celis and D. R. Musicant. Weka-parallel: machine learning in parallel. Technical report, Carleton College, CS TR, 2002.

[112] P. C. G. Costa and K. B. Laskey. Pr-owl: A framework for probabilistic ontologies. In *Proceeding of the 2006 conference on Formal Ontology in Information Systems: Proceedings of the Fourth International Conference (FOIS 2006)*, pages 237–249, Amsterdam, The Netherlands, The Netherlands, 2006. IOS Press.

[113] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 0018-9448.

[114] W. Dargie. The role of probabilistic schemes in multisensor context-awareness. In *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, PERCOMW '07, pages 27–32, Washington, DC, USA, 2007. IEEE Computer Society Press.

[115] S. De and K. Moessner. Ontology-based context inference and query for mobile devices. In *Proceedings of the IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008.*, pages 1–5. IEEE Computer Society Press, Sept. 2008.

[116] Decision Systems Laboratory, University of Pittsburgh. GeNIe & SMILE. `http://genie.sis.pitt.edu/`, 2005-2007.

[117] S. Fortes, K. Frank, and R. Barco. Reducción del coste computacional de la inferencia de contexto bayesiana mediante el uso de rangos de valores dinámicos. In Á. Mediavilla Sánchez et al., editor, *URSI 2009: XXIV Simposium Nacional de la Unión Científica de Radio*, pages 240–243. PUbliCan - Ediciones de la Universidad de Cantabria, Sept. 2009.

[118] K. Frank, N. Kalatzis, I. Roussaki, and N. Liampotis. Challenges for context management systems imposed by context inference. In T. Pfeifer, D. O'Sullivan, and S. Das, editors, *Proceedings of 2009 ACM/IEEE International Conference on Automatic Computing & Co-Located Workshops*, pages 27–33. ACM Digital Library, June 2009.

[119] K. Frank, P. Robertson, S. Fortes Rodriguez, and R. Barco Moreno. Faster Bayesian context inference by using dynamic value ranges. In J. A. Muhtadi and A. Passarella, editors, *Eigth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2010, March 29 - April 2, 2010, Mannheim, Germany, Workshop Proceedings*, pages 50–55, Los Alamitos, CA, USA, Mar. 2010. IEEE Computer Society Press.

[120] K. Frank, P. Robertson, S. McBurney, N. Kalatzis, I. Roussaki, and M. Marengo. A hybrid preference learning and context refinement architecture. In S. M. McBurney and E. Papadopoulou, editors, *PERSIST Workshop on Intelligent Pervasive*

Environments, Proceedings of the Symposium PERSIST Workshop on Intelligent Pervasive Environments, pages 9–15. SSAISB, April 2009.

[121] K. Frank, M. Röckl, P. Gallego Hermann, and M. T. Morillas Vera. Knowledge representation and inference in context-aware computing environments. In J. Lloret Mauri et al., editor, *Proceedings of Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies. UBICOMM08*, pages 89–95. IEEE Computer Society Press, September 2008.

[122] K. Frank, M. Röckl, and T. Pfeifer. Optimizing dynamic composition of bayesian networks for context sensing and inference. In *the 35th Annual IEEE Conference on Local Computer Networks (LCN 2010)*, pages 312–315, Denver, Colorado, USA, 10 2010.

[123] K. Frank, M. Röckl, and T. Pfeifer. Intelligent information dissemination in collaborative, Context-Aware environments. In *8th IEEE International Workshop on Managing Ubiquitous Communications and Services (MUCS 2011)*, pages 287–293, Seattle, USA, 3 2011.

[124] T. Gu, H. K. Pung, and D. Zhang. A Bayesian approach for dealing with uncertain contexts. In *Proceedings of the 2nd International Conference on Pervasive Computing (Pervasive '04)*, volume 176 of *Advances in Pervasive Computing*. Austrian Computer Society, Apr. 2004.

[125] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, Upper Saddle River, NJ, 2nd edition, 1999. ISBN: 978-0132733502.

[126] H. Kim, Y.-J. Cho, and S.-R. Oh. CAMUS: a middleware supporting context-aware services for network-based robots. In *Proceedings of IEEE Workshop on Advanced Robotics and its Social Impacts, 2005*, pages 237–242, June 2005.

[127] M. Kleek and H. E. Shrobe. A practical activity capture framework for personal, lifetime user modeling. In *Proceedings of the 11th international conference on User Modeling*, UM '07, pages 298–302, Berlin, Heidelberg, 2007. Springer-Verlag.

[128] A. Kofod-Petersen and A. Aamodt. Case-based reasoning for situation-aware ambient intelligence: A hospital ward evaluation study. In L. McGinty and D. Wilson, editors, *Case-Based Reasoning Research and Development. 8th International Conference on Case-Based Reasoning, ICCBR 2009. Seattle, WA, USA, July 20-23, 2009. Proceedings*, volume 5650 of *Lecture Notes in Computer Science*, pages 450–464. Springer Berlin / Heidelberg, 2009.

[129] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31:249–268, 2007. Slovenian Society Informatika, ISSN: 0350-5596.

[130] M. Kranz, K. Frank, D. Hermosilla Galceran, and E. Weber. Open vehicular data interfaces for in-car context inference. In A. Schmidt and A. Dey, editors, *Proceedings of 1st International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2009)*, pages 57–62. ACM, Sept. 2009.

[131] G. Malinowski. Many-valued logic and its philosophy. In D. M. Gabbay and J. Woods, editors, *The Many Valued and Nonmonotonic Turn in Logic*, volume 8 of *Handbook of the History of Logic*, pages 13–94. Elsevier Science Publishers B.V., North-Holland, 2007.

[132] M. Perttunen, J. Riekki, and O. Lassila. Context representation and reasoning in pervasive computing: a review. *International Journal of Multimedia and Ubiquitous Engineering*, 4(4):1–28, 2009. Science and Engineering Support Society, ISSN: 1975-0080.

[133] A. Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science, 1977*, pages 46–57, Nov. 1977.

[134] I. Roussaki, N. Liampotis, N. Kalatzis, K. Frank, and P. Hayden. How to make personal smart spaces context-aware. In S. M. McBurney and E. Papadopoulou, editors, *PERSIST Workshop on Intelligent Pervasive Environments*, Proceedings of the Symposium PERSIST Workshop on Intelligent Pervasive Environments, pages 26–32. SSAISB, April 2009.

[135] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NY, USA, 1976. ISBN: 978-0691100425.

[136] T. Strang and C. Linnhoff-Popien, editors. *Location- and Context-Awareness, First International Workshop, LoCA 2005, Oberpfaffenhofen, Germany, May 12-13, 2005, Proceedings*, volume 3479 of *Lecture Notes in Computer Science*. Springer, 2005.

[137] The University of Waitako. Weka 3: Data mining software in java. `http://www.cs.waikato.ac.nz/ml/weka/`, 1993-2008.

[138] X. H. Wang, D. Zhang, T. Gu, and H. K. Pung. Ontology based context modeling and reasoning using OWL. *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004*, pages 18–22, 2004.

[139] H. Wu. *Sensor data fusion for context-aware computing using dempster-shafer theory*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2004. AAI3126933.

8.7 Positioning and Location Management

[140] M. Angermann, J. Kammann, and B. Lami. A new mobilty model based on maps. In *Proceedings of the 58th IEEE Semiannual Vehicular Technology Conference (VTC Fall 2003)*, volume 5, pages 3045–3049. IEEE Computer Society Press, Oct. 2003.

[141] M. Angermann, J. Kammann, P. Robertson, A. Steingaß, and T. Strang. Software representation for heterogeneous data sources within a probabilistic framework. In *Proceedings of International Symposium on Location Based Services for Cellular Users, Munich/Germany*, February 2001.

[142] P. Bahl and V. N. Padmanabhan. Radar: An in-building RF-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, volume 2, pages 775–784. IEEE Computer Society Press, 2000.

[143] S. Beauregard. Omnidirectional pedestrian navigation for first responders. In *Proceedings of the 4th Workshop on Positioning, Navigation and Communication, 2007, WPNC07*, pages 33–36, Hannover, Germany, Mar. 2007.

[144] S. Beauregard, Widyawan, and M. Klepal. Indoor PDR performance enhancement using minimal map information and particle filters. In *Proceedings of IEEE/ION PLANS 2008*, volume VII, pages 141–147. IEEE Computer Society Press, May 2008.

[145] C. Becker and F. Dürr. On location models for ubiquitous computing. *Personal Ubiquitous Computing*, 9:20–31, January 2005. Springer Verlag, London, UK, ISSN: 1617-4909.

[146] J. Collin, O. Mezentsev, and G. Lachapelle. Indoor positioning system using accelerometry and high accuracy heading sensors. In *Proceedings of the 16th International Technical Meeting of the Satellite Division of the Institute of Navigation ION GPS/GNSS 2003*, pages 1164–1170, Portland, OR, USA, Sept. 2003.

[147] COST. Action 231, digital mobile radio towards future generation systems, (final report). In *Office for official publications of the European Communities*, 1999.

[148] S. Domnitcheva. Location modeling: State of the art and challenges. In *Proceedings of the Workshop on Location Modeling for Ubiquitous Computing*, volume 9, pages 13–19, 2001.

[149] F. Evennou and F. Marx. Advanced integration of WIFI and inertial navigation systems for indoor mobile positioning. *EURASIP Journal on Applied Signal Processing*, 2006:164–164, January 2006. Hindawi Publishing Corp., New York, NY, USA, ISSN: 1110-8657.

[150] E. Foxlin. Pedestrian tracking with shoe-mounted inertial sensors. *IEEE Computer Graphics and Applications*, 25(6):38–46, 2005. IEEE Computer Society, Los Alamitos, CA, USA, ISSN: 0272-1716.

[151] K. Frank, B. Krach, N. Catterall, and P. Robertson. Development and evaluation of a combined wlan and inertial indoor pedestrian positioning system. In *Proceedings of the 22nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2009)*, pages 538–546, Sept. 2009.

[152] V. Gabaglio and B. Merminod. Real-time calibration of length of steps with GPS and accelerometers. In *Proceedings of the ENC Global Navigation Satellite System 1999*, volume 2, pages 599–605, Genova, Italia, Oct. 1999.

[153] S. Godha, G. Lachapelle, and M. E. Cannon. Integrated GPS/INS system for pedestrian navigation in a signal degraded environment. In *Proceedings of the 19th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2006)*, pages 2151–2164, Fort Worth, TX, USA, Sept. 2006.

[154] M. F. Goodchild. Twenty years of progress: GIScience in 2010. *Journal of Spatial Information Science*, 1(1):3–20, 2010. University of Maine, USA, ISSN: 1948-660X.

[155] M. Grewal, L. Weill, and A. Andrews. *Global Positioning Systems, Inertial Navigation, and Integration*, volume 5 of *IEE Radar, Navigation and Avionics Series*. John Wiley & Sons, Inc., New York, second edition, 2001. ISBN: 978-0471350323.

[156] P. D. Groves, G. W. Pulford, C. A. Littlefield, D. L. J. Nash, and C. J. Mather. Inertial navigation versus pedestrian dead reckoning: Optimizing the integration. In *Proceedings of the 20th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2007)*, pages 2043—2055, Fort Worth, TX, USA, Sept. 2007.

[157] C. Guggenmos, J. Pérez, D. L. de Ipiña, and G. Gil. A hybrid and semantic location management system for mobile user generated geo services. In *Toulouse Space Show'10*, Toulouse, France, 2010. International Week on Space Applications. Online: `http://www.mugges-fp7.org/pdf/MUGGES-ToulouseSpaceShow.pdf`.

[158] A. Guttman. R-trees: A dynamic index structure for spatial searching. In B. Yormark, editor, *SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984*, pages 47–57. ACM Press, 1984.

[159] H. Hu and D.-L. Lee. Semantic location modeling for location navigation in mobile environment. In *Proceedings of IEEE International Conference on Mobile Data Management, 2004*, pages 52–61, Aug. 2004.

[160] IEEE Std 802.11b-1999. Supplement to IEEE 802.11 standard, higher-speed physical layer extension in the 2.4 GHz band, 1999.

[161] ISO/IEC 8802-11. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, 1999.

[162] M. Khider, S. Kaiser, P. Robertson, and M. Angermann. The effect of maps-enhanced novel movement models on pedestrian navigation performance. In *Proceedings of European Navigation Conference - Global Navigation Satellite Systems (ENC-GNSS 2008)*, pages 1–11, April 2008.

[163] M. Khider, S. Kaiser, P. Robertson, and M. Angermann. A novel movement model for pedestrians suitable for personal navigation. In *ION NTM 2008*, pages 819–827. The Institute of Navigation, 01 2008.

[164] M. Khider, S. Kaiser, P. Robertson, and M. Angermann. Maps and floor plans enhanced 3D movement model for pedestrian navigation. In *Proceedings of the 22nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2009)*, pages 790–802. The Institute of Navigation, Inc, Oct. 2009.

[165] M. Khider, S. Kaiser, P. Robertson, and M. Angermann. A three dimensional movement model for pedestrian navigation. In *European Navigation Conference - Global Navigation Satellite Systems (ENC-GNSS 2009)*, July 2009.

[166] K. Kolomvatsos, V. Papataxiarhis, and V. Tsetsos. Semantic location based services for smart spaces. In Sicilia and Lytras [187], pages 515–525.

[167] A. Kotanen, M. Hännikäinen, H. Leppäkoski, and T. D. Hämäläinen. Positioning with IEEE 802.11b wireless LAN. In *Proceedings on 14th IEEE Conference on Personal, Indoor and Mobile Radio Communications, PIMRC 2003*, volume 3, pages 2218–2222, Sept. 2003.

[168] B. Krach and P. Robertson. Integration of foot-mounted inertial sensors into a bayesian location estimation framework. In *Proceedings of the 5th Workshop on Positioning, Navigation and Communication, WPNC 2008*, pages 55–61, Hannover, Germany, Mar. 2007.

[169] B. Krach and P. Robertson. Cascaded estimation architecture for integration of foot-mounted inertial sensors. In *Proceedings of the 2008 IEEE/ION Position, Location, and Navigation Symposium, PLANS 2008*, pages 112–119, Monterey, CA, USA, May 2008.

[170] D. Kubrak, C. Macabiau, and M. Monnerat. Performance analysis of MEMS based pedestrian navigation systems. In *Proceedings of the 18th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS 2005)*, pages 2976–2986, Long Beach, California, USA, Sept. 2005.

[171] Q. Ladetto. On foot navigation: continuous step calibration using both complementary recursive prediction and adaptive Kalman filtering. In *Proceedings of the 13th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 2000)*, pages 1735–1740, Sept. 2000.

[172] R. Liu and L. Yang. Kernel estimation of multivariate cumulative distribution function. *Journal of Nonparametric Statistics*, 20(8):661–677, 2008. Taylor & Francis, ISSN: 1048-5252.

[173] Y. Malkani and L. Dhomeja. Location aware device discovery for physically constrained environments. In *Proceedings of 2nd International Conference on Computer, Control and Communication, IC4 2009*, pages 1–5, Feb. 2009.

[174] K. L. Myers and D. E. Wilkins. Reasoning about locations in theory and practice. *Computational Intelligence*, 14(2):151–187, 1998. Blackwell Publishers Inc., ISSN: 1467-8640.

[175] A. K. Narayanan. Realms and states: a framework for location aware mobile computing. In *Proceedings of the 1st international workshop on Mobile commerce*, WMC '01, pages 48–54, New York, NY, USA, 2001. ACM.

[176] M. Ocana, L. Bergasa, M. Sotelo, J. Nuevo, and R. Flores. Indoor robot localization system using WiFi signal measure and minimizing calibration effort. In *Proceedings of the IEEE International Symposium on Industrial Electronics, 2005.*, pages 1545–1550, June 2005.

[177] T. Pfeifer. Synergetic positioning architecture for location-dependent services. In *Proceedings of the 29th Annual IEEE Conference on Local Computer Networks, LCN 2004*, pages 404–406, Los Alamitos, CA, USA, Nov. 2004. IEEE Computer Society Press.

[178] T. Pfeifer. Redundant positioning architecture. *Computer Communications*, 28(13):1575–1585, 2005. Wireless Sensor Networks and Applications - Proceedings of the Dagstuhl Seminar 04122. Elsevier Science Publishers B.V., North-Holland, ISSN: 0140-3664.

[179] T. Pfeifer. Autonomic Position Management. In *Proceedings of 1st International Symposium on Wireless Pervasive Computing, 2006*, pages 1–5. IEEE Computer Society Press, 2006.

[180] T. Pfeifer, K. Sullivan, and M. O'Foghlu. Scalability of location sensor data fusion. In *Proceedings of the 31th IEEE Conference on Local Computer Networks, LCN 2006*, pages 565–566, Los Alamitos, CA, USA, Nov. 2006. IEEE Computer Society Press.

[181] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the Sixth Annual ACM International Conference on Mobile Computing and Networking (MOBICOM) 2000*, pages 32–43, New York, NY, USA, Aug. 2000. ACM.

[182] P. Steggles. The Ubisense smart space platform; a Ubisense white paper. Technical report, Ubisense, May 2005.

[183] I. J. Quader, B. Li, W. P. Peng, and A. G. Dempster. Use of fingerprinting in Wi-Fi based outdoor positioning. In *Proceedings of International Global Navigation Satellite Systems Symposium 2007*. IGNSS Society, 2007.

[184] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen. A probabilistic approach to WLAN user location estimation. *International Journal of Wireless Information Networks*, 9(3):155–164, 2002. Springer Verlag, ISSN: 1068-9605.

[185] A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computers & Graphics*, 23(6):893–901, Dec. 1999. Elsevier Science Publishers B.V., North-Holland, ISSN: 0097-8493.

[186] J. Seitz, L. Patiño-Studencka, B. Schindler, S. Haimerl, J. G. Boronat, S. Meyer, and J. Thielecke. Sensor data fusion for pedestrian navigation using WLAN and INS. In D. G. für Ortung und Navigation e.V. (DGON), editor, *Proceedings of DGON-Symposium: Gyro Technology Symposium 2007*, volume 11, pages 1–10, Sept. 2007.

[187] M.-Á. Sicilia and M. D. Lytras, editors. *Metadata and Semantics, Post-proceedings of the 2nd International Conference on Metadata and Semantics Research, MTSR 2007, Corfu Island in Greece, 1-2 October 2007*. Springer, 2009.

[188] D. H. Titterton and J. L. Weston. *Strapdown inertial navigation technology*, volume 5 of *IEE Radar, Navigation and Avionics Series*. Peter Peregrinus Ltd., London, second edition, 2004. ISBN: 978-0863412608.

[189] A. C. Varzi. Spatial reasoning and ontology: Parts, wholes, and locations. In M. Aiello, I. Pratt-Hartmann, and J. van Benthem, editors, *Handbook of Spatial Logics*, chapter 1, pages 945–1038. Springer-Verlag, Berlin, 2007. ISBN: 978-1402055867.

[190] K. Wendlandt, M. Khider, M. Angermann, and P. Robertson. Continuous location and direction estimation with multiple sensors using particle filtering. In *Proceedings of the International Conference in Multisensor Fusion and Integration for Intelligent Systems, MFI 2006*, Library of Congress: 2006930785, pages 92–97. IEEE Computer Society Press, Sept. 2006.

[191] K. Wendlandt, P. Robertson, M. Khider, M. Angermann, and K. Sukchaya. Demonstration of a realtime active-tag RFID, java based indoor localization system using particle filtering. In J. Badram et.al, editor, *Adjunct Proceedings of the 9th International Conference on Ubiquitous Computing, UbiComp 2007*, pages 108–111, September 2007.

[192] W. Widyawan, M. Klepal, and S. Beauregard. A backtracking particle filter for fusing building plans with PDR displacement estimates. In *Proceedings of the 5th Workshop on Positioning, Navigation and Communication, 2008, WPNC08*, Hannover, Germany, Mar. 2008.

[193] O. Woodman and R. Harle. Pedestrian localisation for indoor environments. In *Proceedings of the 10th International Conference on Ubiquitous Computing, UbiComp 2008*, pages 114–123, Seoul, South Korea, Sept. 2008.

[194] M. A. Youssef, A. Agrawala, and A. U. Shankar. WLAN location determination via clustering and probability distributions. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, PERCOM '03, pages 143–150, Washington, DC, USA, 2003. IEEE Computer Society Press.

[195] V. Zarimpas, B. Honary, and M. Darnell. Indoor 802.1x based location determination and real-time tracking. In *Proceedings of the IET International Conference on Wireless, Mobile and Multimedia Networks (ICWMMN 2006)*, pages 1–4, Hang Zhou, China, Nov. 2006. IEEE Computer Society Press.

[196] V. Zarimpas, B. Honary, D. Lund, C. Tanriover, and N. Thanopoulos. Location determination and tracking using radio beacons. In *Proceedings of 6th IEE International Conference on 3G and Beyond*, pages 207–211, The IEE, Savoy Place, London, Nov. 2005. IEEE Computer Society Press.

[197] M. Zündt. *A Distributed Community-Based Location Service Architecture*. PhD thesis, Institute for Communication Networks (LKN) , Technical University of Munich, 2007.

8.8 Activity Recognition

[198] K. Aminian, P. Robert, E. Buchser, B. Rutschmann, D. Hayoz, and M. Depairon. Physical activity monitoring based on accelerometry: validation and comparison with video observation. *Medical & Biological Engineering & Computing*, 37(3):304–308, 1999. Springer Verlag, ISSN: 0140-0118.

[199] L. Bao. Physical Activity Recognition from Acceleration Data under Semi-Naturalistic Conditions. Master Thesis of Massachusetts Institute of Technology, August 2003.

[200] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. In *Proceedings of the Second International Conference on Pervasive Computing, PERVASIVE 2004*, pages 1–17. Springer Verlag, Apr. 2004.

[201] N. Bhargava Bharatula and G. Tröster. On-body context recognition with miniaturized autonomous sensor button. *Technisches Messen*, 74:621–628, Dec. 2007. Oldenbourg Wissenschaftsverlag GmbH, ISSN: 0171-8096.

[202] G. Bieber and C. Peter. Using physical activity for user behavior analysis. In *PETRA '08: Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments*, pages 1–6, New York, NY, USA, 2008. ACM.

[203] S. Bracy, L. E. Dunne, R. Tynan, D. Diamond, B. Smyth, and G. M. P. O'Hare. Garment-based monitoring of respiration rate using a foam pressure sensor. In *ISWC '05: Proceedings of the Ninth IEEE International Symposium on Wearable Computers*, pages 214–215, Washington, DC, USA, 2005. IEEE Computer Society.

[204] G. S. Chambers, S. Venkatesh, G. A. W. West, and H. H. Bui. Hierarchical recognition of intentional human gestures for sports video annotation. In *Proceedings of the 16th International Conference on Pattern Recognition*, volume 2, pages 1082–1085. IEEE Computer Society Press, 2002.

[205] T. Choudhury, G. Borriello, S. Consolvo, D. Haehnel, B. Harrison, B. Hemingway, J. Hightower, P. P. Klasnja, K. Koscher, A. LaMarca, J. A. Landay, L. LeGrand, J. Lester, A. Rahimi, A. Rea, and D. Wyatt. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2):32–41, June 2008. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 1536-1268.

[206] T. Choudhury and A. Pentland. The sociometer: A wearable device for understanding human networks. In *Proceedings of the Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments in cooperation with the Conference on Computer Supported Cooperative Work (CSCW '02)*, Nov. 2002. Online: `http://www.media.mit.edu/~tanzeem/TR-554.pdf`.

[207] L. E. Dunne, P. Walsh, B. Smyth, and B. Caulfield. Design and evaluation of a wearable optical sensor for monitoring seated spinal posture. In *ISWC '06: Proceedings of the Tenth IEEE International Symposium on Wearable Computers*, pages 65–68. IEEE Computer Society Press, 2006.

[208] J. Farringdon, A. J. Moore, N. Tilbury, J. Church, and P. D. Biemond. Wearable sensor badge and sensor jacket for context awareness. In *ISWC '99: Proceedings of the Third IEEE International Symposium on Wearable Computers*, page 107, Washington, DC, USA, 1999. IEEE Computer Society.

[209] K. Frank, M. J. Vera Nadales, P. Robertson, and M. Angermann. Reliable real-time recognition of motion related human activities using mems inertial sensors. In *Proceedings of the 23rd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2010)*, pages 2919–2932, Sept. 2010.

[210] K. Frank, M. J. Vera Nadales, P. Robertson, and T. Pfeifer. Bayesian recognition of motion related activities with inertial sensors. In J. Scott, A. Tabard, J. Bunde-Pedersen, M. B. Kjaergaard, H.-H. Chu, C. Holzmann, A. V. Moore, E. M. Huang, and M. Hazas, editors, *12th ACM International Conference on Ubiquitous Computing*, pages 445–446. Association for Computing Machinery, Inc. (ACM), Sept. 2010.

[211] K. Frank, M. J. Vera Nadales, M. Röckl, and P. Robertson. Comparison of exact static and dynamic bayesian context inference methods for activity recognition. In D. O'Sullivan, T. Pfeifer, and B. Stiller, editors, *MUCS 2010: Proceedings of the 7th IEEE International Workshop on Managing Ubiquitous Communications and Services, March 29, Mannheim, Germany, in cooperation with the 8th Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2010*, number 12 in multicon lecture notes, pages 41–47. Multicon, Schöneiche, Mar. 2010.

[212] D. T. G. Huynh. *Human Activity Recognition with Wearable Sensors*. PhD thesis, Technische Universität Darmstadt, August 2008.

[213] S. S. Intille, L. Bao, E. M. Tapia, and J. Rondoni. Acquiring in situ training data for context-aware ubiquitous computing applications. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1–8, New York, NY, USA, 2004. ACM.

[214] P. Korpipää, M. Koskinen, J. Peltola, S.-M. Mäkelä, and T. Seppänen. Bayesian approach to sensor-based context awareness. *Personal and Ubiquitous Computing*, 7:113–124, 2003. Springer London, ISSN: 1617-4909.

[215] S. Kurata, M. Makikawa, M. Kawato, H. Kobayashi, A. Takahashi, and R. Tokue. Ambulatory physical activity monitoring system. *Studies in Health Technology and Informatics website*, 1998(2):277–281, 1998. IOS Press, ISSN: 0926-9630.

[216] K. V. Laerhoven and O. Cakmakci. What shall we teach our pants? In *ISWC '00: Proceedings of the 4th IEEE International Symposium on Wearable Computers*, pages 77–83, Washington, DC, USA, 2000. IEEE Computer Society Press.

[217] S.-W. Lee and K. Mase. Activity and location recognition using wearable sensors. *IEEE Pervasive Computing*, 1(3):24–32, 2002. IEEE Computer Society Press, Los Alamitos, CA, USA, ISSN: 1536-1268.

[218] J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 766–772, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[219] L. Liao, D. Fox, and H. Kautz. Extracting places and activities from GPS traces using hierarchical conditional random fields. *International Journal of Robotics Research*, 26(1):119–134, 2007. Sage Publications, Inc., Thousand Oaks, CA, USA, ISSN: 0278-3649.

[220] C. Lombriser, N. B. Bharatula, D. Roggen, and G. Tröster. On-body activity recognition in a dynamic sensor network. In *BodyNets '07: Proceedings of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering (ICST) 2nd international conference on Body area networks*, pages 1–6, Brussels, Belgium, 2007. ICST.

[221] M. Losch, S. Schmidt-Rohr, S. Knoop, S. Vacek, and R. Dillmann. Feature set selection and optimal classifier for human activity recognition. In *Proceedings of the 16th IEEE International Symposium on Robot and Human interactive Communication, RO-MAN 2007*, pages 1022–1027. IEEE Computer Society Press, Aug. 2007.

[222] J. Makhoul, F. Kubala, R. Schwartz, and R. Weischedel. Performance measures for information extraction. In *Proceedings of the DARPA Broadcast News Workshop '99*, pages 249–252, San Francisco, CA, USA, July 1999. Morgan Kaufmann Publishers Inc.

[223] M. Makikawa and H. Iizumi. Development of an ambulatory physical activity memory device and its application for the categorization of actions in daily life. *Medinfo*, 2995(2):747–750, 1995. MEDINFO, ISSN: 1091-8140.

[224] J. Mantyjarvi, J. Himberg, and T. Seppanen. Recognizing human motion with multiple acceleration sensors. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, 2001*, volume 2, pages 747–752, 2001.

[225] D. Minnen, T. Starner, I. Essa, and C. Isbell. Discovering characteristic actions from on-body sensor data. In *ISWC '06: Proceedings of the Tenth IEEE International Symposium on Wearable Computers*, pages 11–18, Los Alamitos, AC, USA, 2006. IEEE Computer Society Press.

[226] N. Oliver and F. Flores-Mangas. Healthgear: A real-time wearable system for monitoring and analyzing physiological signals. In *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks*, pages 61–64, Washington, DC, USA, 2006. IEEE Computer Society.

[227] S. Pirttikangas, K. Fujinami, and T. Nakajima. Feature selection and activity recognition from wearable sensors. In H. Y. Youn, M. Kim, and H. Morikawa, editors, *Proceedings of the Third International Symposium on Ubiquitous Computing Systems, UCS 2006*, volume 4239 of *Lecture Notes in Computer Science*, pages 516–527. Springer Verlag, Oct. 2006.

[228] A. Raj, A. Subramanya, D. Fox, and J. A. Bilmes. Rao-blackwellized particle filters for recognizing activities and spatial context from wearable sensors. In O. Khatib, V. Kumar, and D. Rus, editors, *Experimental Robotics, The 10th International Symposium on Experimental Robotics [ISER 06, July 6-10, 2006, Rio de Janeiro, Brazil]*, volume 39 of *Springer Tracts on Advanced Robotics*, pages 211–221. Springer Verlag, 2006.

[229] C. Randell and H. Muller. Context awareness by analyzing accelerometer data. In *ISWC '00: Proceedings of the Fourth IEEE International Symposium on Wearable Computers*, pages 175–176, Washington, DC, USA, 2000. IEEE Computer Society Press.

[230] K. C. Smith. *Bayesian methods for visual multi-object tracking with applications to human activity recognition.* PhD thesis, École Polytechnique Fédérale de Lausanne, Lausanne , Switzerland, 2007. Thèse sciences Ecole polytechnique fédérale de Lausanne EPFL, no 3745 (2007), Faculté des sciences et techniques de l'ingénieur STI, Section de génie électrique et électronique, Institut de génie électrique et électronique IEL (Laboratoire de l'IDIAP). Dir.: Hervé Bourlard, Daniel Gatica-Perez.

[231] T. Stiefmeier. *Real-Time Spotting of Human Activities in Industrial Environments.* PhD thesis, TU Darmstadt, 2008.

[232] T. Stiefmeier, C. Lombriser, D. Roggen, H. Junker, G. Ogris, and G. Tröster. Event-based activity tracking in work environments. In *Proceedings of the 3rd International Forum on Applied Wearable Computing (IFAWC)*, pages 91–100. VDE Verlag, Mar. 2006.

[233] M. Stikic, K. V. Laerhoven, and B. Schiele. Exploring semi-supervised and active learning for activity recognition. In *ISWC '08: Proceedings of the Twelfth IEEE International Symposium on Wearable Computers*, pages 14, 19, 107. IEEE Press, 2008.

[234] A. Subramanya and A. Raj. Recognizing activities and spatial context using wearable sensors. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI'06)*, pages 494–502, Arlington, Virginia, 2006. AUAI Press.

[235] E. M. Tapia, S. S. Intille, and K. Larson. Activity recognition in the home using simple and ubiquitous sensors. In A. Ferscha and F. Mattern, editors, *Pervasive Computing. Second International Conference, PERVASIVE 2004, Vienna Austria, April 21-23, 2004, Proceedings*, volume 3001 of *Lecture Notes in Computer Science*, pages 158–175. Springer Verlag, 2004.

[236] M. Uiterwaal, E. Glerum, H. Busser, and R. van Lummel. Ambulatory monitoring of physical activity in working situations, a validation study. *Journal of Medical Engineering & Technology*, 22(4):168–172, July 1998. Taylor & Francis, ISSN: 0309-1902.

[237] M. J. Vera Nadales. Recognition of human motion related activities from sensors. Master's thesis, University of Malaga, 2010.

[238] E. Welbourne, J. Lester, A. LaMarca, and G. Borriello. Mobile context inference using low-cost sensors. In *Location- and Context-Awareness: First International Workshop, LoCA 2005, Oberpfaffenhofen, Germany, May 12-13, 2005, Proceedings*, volume 3479 of *Lecture Notes in Computer Science*, pages 254–263. Springer Verlag, 2005.

[239] T. L. Westeyn, P. Presti, and T. Starner. ActionGSR: A combination galvanic skin response-accelerometer for physiological measurements in active environments. In *ISWC '06: Proceedings of the Tenth IEEE International Symposium on Wearable Computers*, pages 129–130. IEEE Computer Society Press, 2006.

[240] D. A. Winter. *Biomechanics and Motor Control of Human Movement*. Wiley, August 2004. ISBN: 978-0471509080.

[241] XSens Technologies B.V. *MTi and MTx User Manual and Technical Documentation*, 2007. Online: `http://eris.liralab.it/viki/images/8/82/XsensMtx.pdf`.

[242] J.-Y. Yang, Y.-P. Chen, G.-Y. Lee, S.-N. Liou, and J.-S. Wang. Activity recognition using one triaxial accelerometer: A neuro-fuzzy classifier with feature reduction. In L. Ma, M. Rauterberg, and R. Nakatsu, editors, *Entertainment Computing - ICEC 2007. 6th International Conference, Shanghai, China, September 15-17, 2007, Proceedings*, volume 4740 of *Lecture Notes in Computer Science*, pages 395–400. Springer Verlag, 2007.

[243] P. Zappi, C. Lombriser, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Tröster. Activity recognition from on-body sensors: Accuracy-power trade-off by dynamic sensor selection. In R. Verdone, editor, *EWSN'08. Proceedings of the 5th European conference on Wireless sensor networks*, volume 4913 of *Lecture Notes in Computer Science*, pages 17–33, Berlin Heidelberg, 2008. Springer Verlag.

8.9 Tractability in Context Awareness and Bayesian Techniques

[244] O. Bangsø and P.-H. Wuillemin. Object oriented Bayesian networks: A framework for topdown specification of large Bayesian networks and repetitive structures. Technical report, Department of Computer Science, Aalborg University, Aalborg, Denmark, 2000. Online: `http://citeseer.ist.psu.edu/viewdoc/download?doi=10.1.1.33.5490&rep=rep1&type=pdf`.

[245] O. Bangsø. *Object Oriented Bayesian Networks*. PhD thesis, Aalborg University, Department of Computer Science, 2004.

[246] O. Bangsø, M. J. Flores, and F. V. Jensen. Plug & play object oriented bayesian networks. In *Proceedings of the Tenth Conference of the Spanish Association for Artificial Intelligence (CAEPIA)*, volume 3040, pages 457–467. Springer Verlag, 2003.

[247] C. Barratt, A. Brogni, M. Chalmers, W. R. Cobern, J. Crowe, D. Cruickshank, N. Davies, D. D. Roure, A. Friday, A. Hampshire, O. J. Gibson, C. Greenhalgh,

B. Hayes-Gill, J. Humble, H. Muller, B. Palethorpe, T. Rodden, C. Setchell, M. Sumner, O. Storz, and L. Tarassenko. Extending the Grid to Support Remote Medical Monitoring. In S. J. Cox, editor, *Proceedings of the 2nd UK E-Science All Hands Meeting, EPSRC*, pages 563–570. University of Southampton, Sept. 2003.

[248] R. B'Far. Scalable reasoning techniques for semantic enterprise data. In D. Wood, editor, *Linking Enterprise Data*, pages 127–147. Springer US, 2010. ISBN: 978-1-4419-7664-2.

[249] T. Buchholz and C. Linnhoff-Popien. Towards realizing global scalability in context-aware systems. In T. Strang and C. Linnhoff-Popien, editors, *Location- and Context-Awareness: First International Workshop, LoCA 2005, Oberpfaffenhofen, Germany, May 12-13, 2005, Proceedings*, volume 3479 of *Lecture Notes in Computer Science*, pages 15–17, Berlin / Heidelberg, 2005. Springer Verlag.

[250] M. J. Flores, J. A. Gámez, and K. G. Olesen. Incremental compilation of Bayesian networks. In C. Meek and U. Kjærulff, editors, *UAI '03: Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence*, pages 233–240. Morgan Kaufmann, Aug. 2003.

[251] M. J. Flores Gallego. *Bayesian networks Inference: Advanced algorithms for triangulation and partial abduction.* PhD thesis, Universidad de Castilla-La Mancha. Departamento de Sistemas Informáticos., 2005.

[252] D. Geiger and P. P. Shenoy, editors. *UAI '97: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, August 1-3, 1997, Brown University, Providence, Rhode Island, USA*. Morgan Kaufmann, 1997.

[253] K.-S. Hwang and S.-B. Cho. Modular Bayesian networks for inferring landmarks on mobile daily life. In *AI 2006: Advances in Artificial Intelligence, 19th Australian Joint Conference on Artificial Intelligence, Hobart, Australia, December 4-8, 2006, Proceedings*, volume 4304 of *Lecture Notes in Computer Science*, pages 929–933. Springer Verlag, Dec. 2006.

[254] K.-S. Hwang and S.-B. Cho. Modular Bayesian network learning for mobile life understanding. In *Proceedings of the 9th International Conference on Intelligent Data Engineering and Automated Learning*, IDEAL '08, pages 225–232, Berlin, Heidelberg, 2008. Springer Verlag.

[255] K.-S. Hwang and S.-B. Cho. Modular Bayesian network learning for uncertainty handling on mobile device. In M. Ojeda-Aciego, L. Magdalena, and J. Verdegay, editors, *Proceedings of the 12th International Conference on Information Processing and Management of Unvertainty in Knowledge-Based Systems, IPMU'08*, pages 402–408, June 2008.

[256] K. Jean, A. Galis, and A. Tan. Context-aware GRID services: Issues and approaches. In M. Bubak, G. D. v. Albada, P. M. A. Sloot, and J. J. Dongarra, editors, *Computational Science - ICCS 2004: 4th International Conference, Kraków, Poland, June 6-9, 2004, Proceedings*, volume 3038 of *Lecture Notes in Computer Science*, pages 166–173. Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-24688-6_24.

[257] D. Koller and A. Pfeffer. Object-oriented Bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97)*, pages 302–313. Morgan Kaufmann Publishers Inc., Aug. 1997.

[258] K. B. Laskey and S. M. Mahoney. Network fragments: Representing knowledge for constructing probabilistic models. In Geiger and Shenoy [252], pages 334–341.

[259] B. Motik and U. Sattler. A comparison of reasoning techniques for querying large description logic ABoxes. In *Proceedings of the 13th International Conference on Logic for Programming Artificial Intelligence and Reasoning LPAR 2006*, pages 227–241. Springer Verlag, 2006.

[260] B. C. Neuman. Scale in distributed systems. In T. L. Casavant and M. Singhal, editors, *Readings in Distributed Computing Systems*, pages 463–489. IEEE Computer Society Press, Los Alamitos, CA, USA, 1994. ISBN: 0818630329.

[261] K. G. Olesen and A. L. Madsen. Maximal prime subgraph decomposition of Bayesian networks. *IEEE Transactions on Systems, Man and Cybernetics: Part B*, 32(1):21–31, Feb. 1999. IEEE Computer Society Press, New York, NY, USA, ISSN: 1083-4419.

[262] P. d. Oude and G. Pavlin. An information theoretic approach to verification of modular Bayesian fusion systems. In *Proceedings of the 11th International Conference on Information Fusion*, pages 1–8. IEEE Computer Society Press, June 2008.

[263] P. d. Oude and G. Pavlin. Constraint-based approach to discovery of inter module dependencies in modular Bayesian networks. In H. C. Lane and H. W. Guesgen, editors, *Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference*, pages 1–6. AAAI Press, May 2009.

[264] P. d. Oude and G. Pavlin. Efficient distributed bayesian reasoning via targeted instantiation of variables. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02*, WI-IAT '09, pages 323–330, Washington, DC, USA, 2009. IEEE Computer Society.

[265] P. d. Oude, G. Pavlin, and T. Hood. A modular approach to adaptive Bayesian information fusion. In *Proceedings of the 10th International Conference on Information Fusion, Fusion 2007*, pages 1–8. IEEE Computer Society Press, July 2007.

[266] M. A. Paskin and C. E. Guestrin. Robust probabilistic inference in distributed systems. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, pages 436–445, Arlington, Virginia, USA, 2004. AUAI Press.

[267] G. Pavlin, P. d. Oude, M. Maris, and T. Hood. Distributed perception networks: An architecture for information fusion systems based on causal probabilistic models. In *Proceedings of the IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 303–310. IEEE Computer Society Press, 2006.

[268] M. Salvadores, G. Correndo, T. Omitola, N. Gibbins, S. Harris, and N. Shadbolt. 4s-reasoner: RDFS backward chained reasoning support in 4store. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 261–264, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[269] O. Storz, A. Friday, and N. Davies. Towards 'ubiquitous' ubiquitous computing: an alliance with the grid. In *Proceedings of the First Workshop on System Support for Ubiquitous Computing Workshop (Ubisys 2003) in association with Fifth International Conference on Ubiquitous Computing*, Oct. 2003.

[270] H. Tu, J. Allanach, S. Singh, K. R. Pattipati, and P. Willett. Information integration via hierarchical and hybrid Bayesian networks. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 36(1):19–33, Jan. 2006. IEEE Computer Society Press, ISSN: 1083-4427.

[271] Y. Xiang. *Probabilistic Reasoning in Multi-Agent Systems: A Graphical Models Approach.* Cambridge University Press, New York, NY, USA, 2002. ISBN: 0521813085.

[272] Y. Xiang and F. V. Jensen. Inference in multiply sectioned Bayesian networks with extended shafer-shenoy and lazy propagation. In K. B. Laskey and H. Prade, editors, *UAI '99: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 680–687. Morgan Kaufmann Publishers Inc., July 1999.

[273] Y. Xiang, D. Poole, and M. P. Beddoes. Multiply sectioned Bayesian networks and junction forests for large knowledge-based systems. *Computational Intelligence*, 9(2):171–220, 1993. Blackwell Publishing Ltd., ISSN: 1467-8640.

List of the Author's Publications

[SLPF03a] T. Strang, C. Linnhoff-Popien, and K. Frank. Applications of a context ontology language. In *Proceedings of the 11th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2003)*, pages 14–18, October 2003.

[SLPF03b] T. Strang, C. Linnhoff-Popien, and K. Frank. Cool: A context ontology language to enable contextual interoperability. In J.-B. Stefani, I. Demeure, and D. Hagimont, editors, *4th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems (DAIS 2003)*, volume LNCS 2893 of *Lecture Notes in Computer Science*, pages 236–247. Springer-Verlag, Heidelberg, November 2003. Cited in Reference [42].

[SLPKF03] T. Strang, C. Linnhoff-Popien, and I. Korbinian Frank. Integration issues of an ontology based context modelling approach. In P. Isaias and N. Karmakar, editors, *Proceedings of the IADIS International Conference WWW/Internet 2003*, pages 361–368, November 2003.

[RFRS06a] M. Röckl, K. Frank, P. Robertson, and T. Strang. Improved information processing for cooperative vehicle safety applications. In G. für Verkehr Braunschweig e.V., editor, *Informationssysteme für mobile Anwendungen (IMA 2006)*, pages 204–213, October 2006.

[RFRS06b] M. Röckl, K. Frank, P. Robertson, and T. Strang. Enhancing driver assistance systems by cooperative situation awareness. In D. G. für Ortung und Navigation e.V. (DGON), editor, *Proceedings of DGON-Symposium: System Verkehr - Steuern, Regeln, Entwickeln*, November 2006.

[RRFS07] M. Röckl, P. Robertson, K. Frank, and T. Strang. An architecture for situation-aware driver assistance systems. In *IEEE 65th Vehicular Technology Conference - Spring 2007*, pages 2555–2559. IEEE Computer Society Press, April 2007.

[FSM08] K. Frank, V. Suraci, and J. Mitic. Personalizable service discovery in pervasive systems. In Bi Jun et al., editor, *Proceedings of the Fourth International Conference on Networking and Services. ICNS2008*, pages 182–187. IEEE Computer Society Press, March 2008.

[Fra08] K. Frank, editor. *Personal Self-Improving Smart Spaces: Scenario description and requirements specification*, Deliverable D2.1. ICT Persist Consortium, July 2008.

[FRGM08] K. Frank, M. Röckl, P. Gallego Hermann, and M. T. Morillas Vera. Knowledge representation and inference in context-aware computing environments.

In J. Lloret Mauri et al., editor, *Proceedings of Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies. UBICOMM08*, pages 89–95. IEEE Computer Society Press, September 2008. Cited in Reference [121].

[FRR08] K. Frank, M. Röckl, and P. Robertson. The Bayeslet concept for modular context inference. In J. Lloret Mauri et al., editor, *Proceedings of Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies. UBICOMM08*, pages 96–101. IEEE Computer Society Press, September 2008.

[RFS+08] M. Röckl, K. Frank, T. Strang, M. Kranz, J. Gacnik, and J. Schomerus. Hybrid fusion approach combining autonomous and cooperative detection and ranging methods for situation-aware driver assistance systems. In *Proceedings of the IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2008*. IEEE Computer Society Press, September 2008.

[FRM+09] K. Frank, P. Robertson, S. McBurney, N. Kalatzis, I. Roussaki, and M. Marengo. A hybrid preference learning and context refinement architecture. In S. M. McBurney and E. Papadopoulou, editors, *PERSIST Workshop on Intelligent Pervasive Environments*, Proceedings of the Symposium PERSIST Workshop on Intelligent Pervasive Environments, pages 9–15. SSAISB, April 2009. Cited in Reference [120].

[RLK+09] I. Roussaki, N. Liampotis, N. Kalatzis, K. Frank, and P. Hayden. How to make personal smart spaces context-aware. In S. M. McBurney and E. Papadopoulou, editors, *PERSIST Workshop on Intelligent Pervasive Environments*, Proceedings of the Symposium PERSIST Workshop on Intelligent Pervasive Environments, pages 26–32. SSAISB, April 2009. Cited in Reference [134].

[FKRL09] K. Frank, N. Kalatzis, I. Roussaki, and N. Liampotis. Challenges for context management systems imposed by context inference. In T. Pfeifer, D. O'Sullivan, and S. Das, editors, *MUCS '09. Proceedings of the 6th international workshop on Managing ubiquitous communications and services in cooperation with the ACM/IEEE International Conference on Automatic Computing & Co-Located Workshops*, pages 27–33, New York, NY, USA, June 2009. ACM Digital Library. Cited in Reference [118].

[FFB09] S. Fortes, K. Frank, and R. Barco. Reducción del coste computacional de la inferencia de contexto bayesiana mediante el uso de rangos de valores dinámicos. In Á. Mediavilla Sánchez et al., editor, *URSI 2009: XXIV Simposium Nacional de la Unión Científica de Radio*, pages 240–243. PUbliCan - Ediciones de la Universidad de Cantabria, September 2009. Cited in Reference [117].

[FKCR09] K. Frank, B. Krach, N. Catterall, and P. Robertson. Development and evaluation of a combined wlan and inertial indoor pedestrian positioning system. In *Proceedings of the 22nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2009)*, pages 538–546, September 2009. Cited in Reference [151].

[KFHW09] M. Kranz, K. Frank, D. Hermosilla Galceran, and E. Weber. Open vehicular data interfaces for in-car context inference. In A. Schmidt and A. Dey, editors, *Proceedings of 1st International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2009)*, pages 57–62. ACM, September 2009. Cited in Reference [130].

[CTW+09] M. Crotty, N. Taylor, H. Williams, K. Frank, I. Roussaki, and M. Roddy. A pervasive environment based on personal self-improving smart spaces. In H. Gerhäuser, J. Hupp, C. Efstratiou, and J. Heppner, editors, *Proceedings of the workshop on architectures and platforms for AmI, AmI08, in cooperation with the European conference on Ambient Intelligence*, volume 32 of *Communications in Computer and Information Science*, pages 58–62. Springer Berlin Heidelberg, 2009.

[FRFB10] K. Frank, P. Robertson, S. Fortes Rodriguez, and R. Barco Moreno. Faster Bayesian context inference by using dynamic value ranges. In J. A. Muhtadi and A. Passarella, editors, *Eigth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2010, March 29 - April 2, 2010, Mannheim, Germany, Workshop Proceedings*, pages 50–55, Los Alamitos, CA, USA, March 2010. IEEE Computer Society Press. Cited in Reference [119].

[FVRR10] K. Frank, M. J. Vera Nadales, M. Röckl, and P. Robertson. Comparison of exact static and dynamic bayesian context inference methods for activity recognition. In D. O'Sullivan, T. Pfeifer, and B. Stiller, editors, *MUCS 2010: Proceedings of the 7th IEEE International Workshop on Managing Ubiquitous Communications and Services, March 29, Mannheim, Germany, in cooperation with the 8th Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2010*, number 12 in multicon lecture notes, pages 41–47. Multicon, Schöneiche, March 2010. Cited in Reference [211].

[MBM+10] T. Mota, N. Baker, B. Moltchanov, I. Roussaki, and K. Frank. Towards pervasive smart spaces: A tale of two projects. In *Proceedings of Future Network & Mobile Summit 2010*, June 2010.

[FVRA10] K. Frank, M. J. Vera Nadales, P. Robertson, and M. Angermann. Reliable real-time recognition of motion related human activities using mems inertial sensors. In *Proceedings of the 23rd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2010)*, pages 2919–2932, September 2010. Cited in Reference [209].

[FVRP10] K. Frank, M. J. Vera Nadales, P. Robertson, and T. Pfeifer. Bayesian recognition of motion related activities with inertial sensors. In J. Scott, A. Tabard, J. Bunde-Pedersen, M. B. Kjaergaard, H.-H. Chu, C. Holzmann, A. V. Moore, E. M. Huang, and M. Hazas, editors, *12th ACM International Conference on Ubiquitous Computing*, pages 445–446. Association for Computing Machinery, Inc. (ACM), September 2010. Cited in Reference [210].

[FRP10] K. Frank, M. Röckl, and T. Pfeifer. Optimizing dynamic composition of bayesian networks for context sensing and inference. In *Proceedings of the 35th Annual IEEE Conference on Local Computer Networks (LCN 2010)*, pages

312–315. IEEE Computer Society Press, October 2010. Cited in Reference [122].

[FRP11] K. Frank, M. Röckl, and T. Pfeifer. Intelligent information dissemination in collaborative, Context-Aware environments. In *Proceedings of the 8th IEEE International Workshop on Managing Ubiquitous Communications and Services (MUCS 2011)*, pages 287–293. IEEE Computer Society Press, March 2011. Cited in Reference [123].

[RKL+11] I. Roussaki, N. Kalatzis, N. Liampotis, K. Frank, E. Sykas, and M. Anagnostou. Developing context-aware personal smart spaces. In P. Alencar and D. Cowan, editors, *Handbook of Research on Mobile Software Engineering: Design, Implementation and Emergent Applications*, volume 1, Hershey, Pennsylvania, USA, July 2011. IGI Global Publisher.

List of Figures

List of Tables

List of Acronyms

6DOF Six Degree Of Freedom

AAL Ambient Assisted Living

ADL Activity of Daily Life

ASCi-Model Aspect-Scale-ContextInformation Model

BF Body Frame

BN Bayesian Network

BN Bayesian network

BPF Band Pass Filter

CACC Cooperative Adaptive Cruise Control

CAS Context aware service

CIR Context Inference Rule

CMS Context Management System

CPD Conditional Probability Distribution

CPT Conditional Probability Table

DA Driver Assistance

DAG Directed Acyclic Graph

DLR Deutsches Zentrum für Luft- und Raumfahrt, the German Aerospace Center

DVR Dynamic Value Range

EKF Extended Kalman Filter

EMD Entropy Minimisation Discretisation

EM Expectation Maximisation

ENS Equal Number of States

GF Global Frame

GNSS Global Navigation Satellite System

GPS Global Positioning System

ICT International Telecommunication Union

IMU Inertial Measurement Unit

INS Inertial Navigation System

IQR Inter-Quartile Range

ITU Information and Communication Technology

JPD Joint Probability Distribution

kNN k-Nearest-Neighbour

LPF Low Pass Filter

MBR Minimum Bounding Rectangle

MCMC Markov Chain Monte Carlo

MDL Minimum Description Length

MEMS Micro Machined Electro-Mechanical Systems

MFC Main Frequency Component

MMP Minimum Merged Probability

MPE Most Probable Explanation

MSBN Multiply sectioned Bayesian networks

OOBN Object Oriented Bayesian Network

PDF Probability Density Function

PDR Pedestrian Dead Reckoning

PE Protection Extension for states of interest

RDF Resource Description Framework

RFID Radio Frequency Identification

RMS Root Mean Square

RSSI Received Signal Strength Indicator

RV Random Variable

SVM Support Vector Machine

UWB Ultra Wideband

V2V Vehicle-to-Vehicle Communications

VR Value Range

XML eXtensible Markup Language

ZUPT Zero Update

Index

Appendix A

Bayesian Network Modelling a User's "Situation"

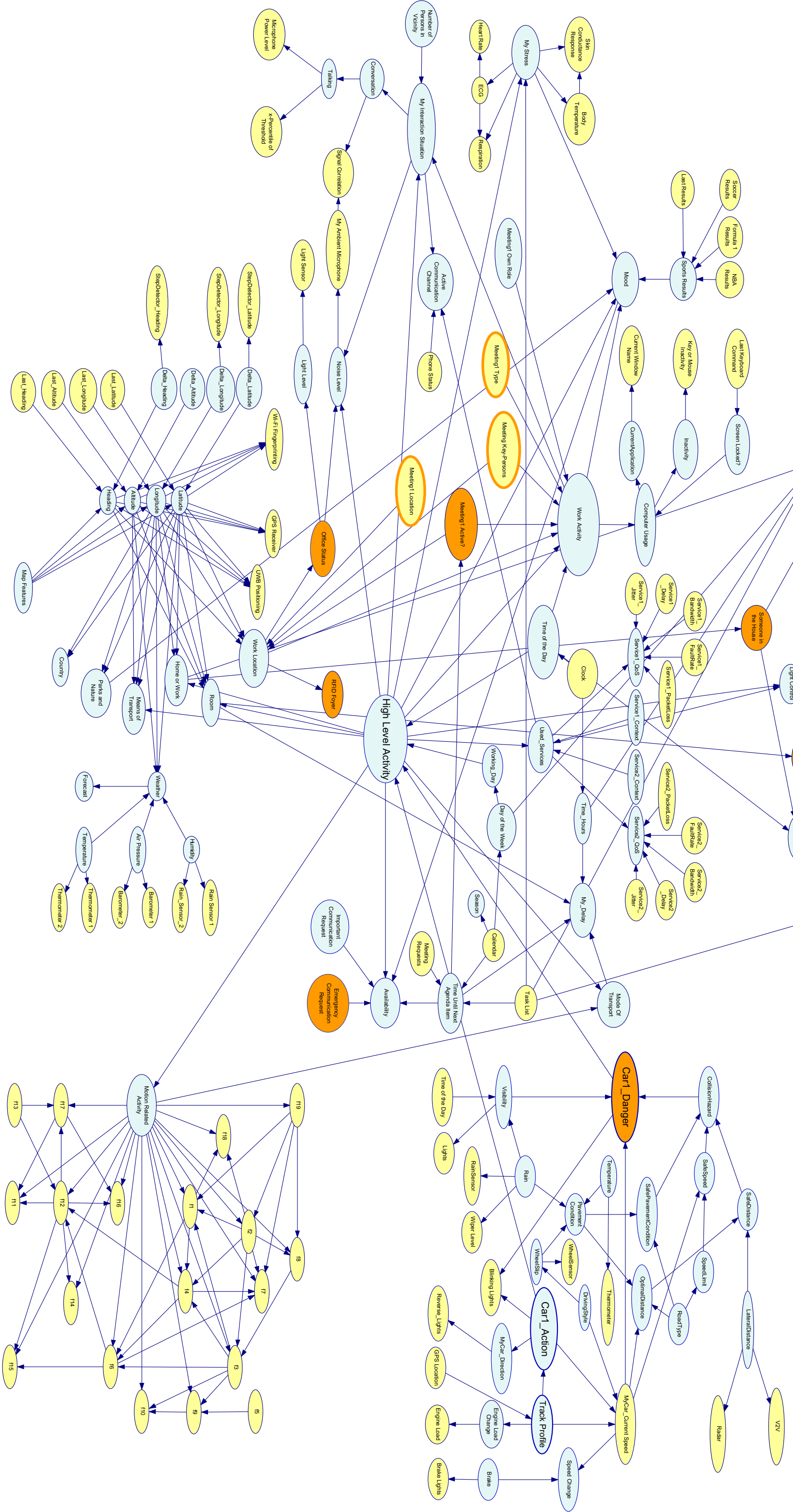
This appendix shows the example Bayesian network used to illustrate the tractability advantages in section 6.5.

The Bayesian network describes many aspects of the situation of a user and is formed from all Bayesian networks used in the previous chapters. Some further information is added which either extends, connects or complements the already shown parts of a user's situation model. Thereby it is still a heavy simplification of the reality. Some parts of the BN are simplified and connections to other entities neglected. It considers only one home, one car, one meeting, and two active services, while it misses the modelling of other cars in traffic and other persons.

The resulting *Situation* network includes 180 random variables, some with discrete, others with continuous value ranges, connected by 275 edges. It can be grouped into more than 50 Bayeslets in which 80 RVs have been marked as input nodes, i.e. have direct input from sensors or other information sources.

The figure shows input nodes in yellow. Nodes receiving input from other, not modelled entities are orange or, if observed, have orange borders. For every connected entity, a BN of about the same size would have to be co-evaluated.

Plate A.1 See next page: a Bayesian network modelling parts of the situation of a user. The model with 180 random variables and 275 edges is still grossly simplified. A grouped version of this Plate is given in Figure 6.23 on page 187.


Car1_Danger
Car1_Action
High Level Activity
Work Activity
Work Location
Computer Usage
My Interaction Situation
My Stress
Mood
Sports Results
Last Results
Soccer Results
Formula 1 Results
NBA Results
Heart Rate
ECG
Respiration
Skin Conductance Response
Body Temperature
Number of Persons in Vicinity
Conversation
Talking
Microphone Power Level
x-Percentile of Threshold
Signal Correlation
My Ambient Microphone
Light Sensor
Light Level
Noise Level
Active Communication Channel
Phone Status
Meeting1 Own Role
Meeting1 Type
Meeting Key-Persons
Meeting1 Location
Meeting1 Active?
Office Status
RFID Foyer
Current Window Name
CurrentApplication
Key or Mouse Inactivity
Inactivity
Last Keyboard Command
Screen Locked?
Fridge Control
Food Necessary
Fridge Filling Status
Home Heating Control
Room Temperature
Room Thermometer
Outdoor Temperature
Outdoor Thermometer
Someone in the House
Someone in the Room
Light Control
Room Light Level
Room Light Sensor
Outdoor Light Level
Outdoor Light Sensor
Clearing Timer
Dust Sensor
Visitation imminent
Used_Services
Service1_Context
Service1_QoS
Service1_ Jitter
Service1 _Delay
Service1_ Bandwidth
Service1_ FaultRate
Service1_PacketLoss
Service2_Context
Service2_QoS
Service2_ Jitter
Service2 _Delay
Service2_ Bandwidth
Service2_ FaultRate
Service2_PacketLoss
Time of the Day
Clock
Working_Day
Day of the Week
Time_Hours
My_Delay
Season
Calendar
Important Communication Request
Emergency Communication Request
Meeting Requests
Availability
Time Until Next Agenda Item
Task List
Mode Of Transport
Weather
Forecast
Temperature
Thermometer 1
Thermometer 2
Air Pressure
Barometer 1
Barometer_2
Humidity
Rain_Sensor_2
Rain Sensor 1
Parks and Nature
Means of Transport
Home or Work
Room
Country
Map Features
Heading
Altitude
Longitude
Latitude
Last_Heading
Last_Altitude
Last_Longitude
Last_Latitude
Delta_Heading
Delta_Altitude
Delta_Longitude
Delta_Latitude
StepDetector_Heading
StepDetector_Longitude
StepDetector_Latitude
Wi-Fi Fingerprinting
GPS Receiver
UWB Positioning
Motion Related Activity
f1
f2
f3
f4
f5
f6
f7
f8
f9
f10
f11
f12
f13
f14
f15
f16
f17
f18
f19
CollisionHazard
SafeDistance
SafeSpeed
SafePavementCondition
SpeedLimit
LateralDistance
OptimalDistance
RoadType
Radar
V2V
Visibility
Time of the Day
Lights
Rain
RainSensor
Wiper Level
Temperature
Pavement Condition
WheelSlip
WheelSensor
Blinking Lights
Reverse_Lights
GPS Location
MyCar_Direction
DrivingStyle
Thermometer
Track Profile
Engine Load Change
Engine Load
Brake
Brake Lights
Speed Change
MyCar_Current Speed