DIPLOMA THESIS

Algorithms for Sensor Fusion in Ambient Assisted Living Systems

Submitted at the Faculty of Electrical Engineering and Information Technology,
Vienna University of Technology
in partial fulfillment of the requirements for the degree of
Diplom-Ingenieur (equals Master of Sciences)

under supervision of

O.Univ.Prof. Dipl.-Ing. Dr.techn. Dietmar Dietrich
Univ.Ass. Dipl.-Ing. Dr.techn. Dietmar Bruckner
Projektass. Dipl.-Ing. Guo Qing Yin

Armin Faltinger
Matr.Nr. 0525788
Kreuzberg 289, 3920 Gross Gerungs

November 17, 2011 _____

Kurzfassung

Ambient Assisted Living-Systeme unterstützen Personen in ihrem Alltag um nicht auf fremde Hilfe angewiesen zu sein, zum Beispiel durch Überwachung des Gesundheitszustands.
Ein gezielte Verhaltensanalyse und in weiterer Folge eine Gesundheitszustandsüberwachung ist nur dann möglich, wenn die Sensorwerte verlässlich sind. Diese Diplomarbeit stellt eine Verbindung zwischen Ambient Assisted Living-Systemen und Sensor Fusion her. Sensor Fusion stellt gängige Methoden zur Verfügung, um Sensordaten verschiedensten Ursprungs miteinander zu kombinieren.
Ein Implementierungsbereich beschäftigt sich mit der Verarbeitung von den Rohdaten der Sensoren. Dabei wurden diverse Algorithmen verwendet, die auch den Aspekt der Fehlertoleranz mit einbeziehen. Um die beste Variante herauszufinden, wurden Bewertungskriterien definiert.
Diese Arbeit liefert auch Lösungen, um die Konfiguration eines solchen Systems zu erleichtern und damit auch die Verwendbarkeit und in weiterer Folge die Akzeptanz zu verbessern, indem die Sensortypen des Knotens und auch die Topologie automatische gelernt werden. Eine Beurteilung der Bewegungsdaten ist dann möglich.
Die Analyse verschiedenster Algorithmen zur Fusion von Rohdaten zeigte, dass sich ein Algorithmus (congeneric multi-sensor data fusion algorithm) besonders gut für diese Aufgabe eignet. Bei der Erkennung der verschiedenen Sensortypen werden hohe Klassifikationsraten (durchschnittlich 96 Prozent) erzielt. In den beobachteten Fällen können die fehlerhaften Klassifikationen immer mit Hilfe von Sensors Fusion durch die Korrekten ersetzt werden. Diese Methode lässt es auch zu, auf die Anwesenheit einer Person zu schließen. Die Feststellung der Topologie ist im Wesentlichen von externen Einflüssen abhängig.
Die vorgestellten Algorithmen dienen als Basis für eine weiterführende Verhaltensanalyse. Die Klassifizierung der Sensoren eines Knoten und das Lernen der Topologie können auch im Bereich der Gebäudeautomation zur Verringerung des Konfigurationsaufwands beitragen.

Abstract

Ambient Assisted Living systems are supporting persons in an independent living by e.g. monitoring their health status
A distinct behavior analysis and in further terms the health monitoring is only possible if reliable sensor data are available. This thesis links the field of Ambient Assisted Living with Sensor Fusion. Sensor Fusion offers common methods to combine various data from different sources.
One part of the implementation is dealing with the processing of raw sensor data. Therefore, the aspect of fault tolerance is taken into account. The most applicable algorithm is found by applying evaluation criteria.
The thesis is also offering some solutions to simplify the installation phase of an Ambient Assisted Living system. The gained benefits are the increased usability and a better acceptance in the long run by automatically learning the sensor types and the topology. The topology identifies connected areas of a flat and is used to evaluate the accuracy of the position data.
The evaluation of algorithms fusing raw data showed that one algorithm (congeneric multi-sensor data fusion algorithm) is practicable for this task. High classification rates (on average 96 percent) can be achieved when identifying the different sensor types. Via Sensor Fusion all incorrect classifications can be replaced by the correct ones. This method can also be used to deduce if a person is present in the flat on a specific day. The determined topology mainly depends on external influences.
The basis for further behavior analysis is provided by the presented algorithms. The node classification process as well as the topology learning algorithm can also be implemented in another context. Sensor classification and learning of the flat's topology may also be applied to building automation in order to reduce configuration work.

Table of Contents

Abbreviations

AAL	Ambient Assisted Living
ATTEND	AdapTive scenario recogniTion for Emergency and Need Detection
CSV	Comma-Separated Values
DAI	Data In
DAO	Data Out
DEI	Decision In
DEO	Decision Out
ECG	Electrocardiogram
EEG	Electroencephalogram
EMG	Electromyography
FEI	Feature In
FEO	Feature Out
GPS	Global Positioning System
HCI	Human Computer Interaction
HCU	e-Home Central Unit
HMI	Human Machine Interaction
HMM	Hidden Markov Model
ICT	Information and Communication Technology
JDL	Joint Directors of Laboratories
JP	Joint Program
LUI	Local User Interface
MAR	Multivariate Autoregressive
MPLS	Mobile Phone Location Services
OODA	Observe-Orientate-Decide-Act
PF2	ProFusion2
PIR	Passive Infra-red
R&D	Research & Development
SF	Sensor Fusion
SME	Small and Medium Sized Enterprise
SNR	Signal-to-noise ratio
TFEU	Treaty on the Functioning of the European Union
TP	Test Person
WIFI	Wireless Fidelity

1 Introduction

The work is positioned in the fields of Ambient Assisted Living and Sensor Fusion. Ambient Assisted Living (AAL) systems are including home automation, monitoring and behaviour analysis functions that support elderly people in an autonomous living. The main focus of this thesis lies in the integration of Sensor Fusion algorithms into AAL systems. The goal of Sensor Fusion is to combine data from various sensors to a more complete representation [SBW99]. Moreover, Sensor Fusion aims to overcome the shortcomings of single sensor systems (see Section 4.1.2).

The first Sections introduces the reader to the topic of Ambient Assisted Living. Therefore, current trends, applications, and issues in respect to the problems of such systems are emphasized. The first examination of Ambient Assisted Living provides the basic thoughts for positioning this work and for discussing the contribution from the thesis to the mentioned field in a later step.

1.1 Ambient Assisted Living

The portion of elderly people will be the most increasing section of the population over the next decades. Due to this effect the age pyramid shifts from a younger to an elderly population [1]. This trend can be seen in Europe as well as in other developed regions. This will cause some major social and economic challenges. Ambient Assisted Living is tackling these problems by supporting elderly people with electronic systems in their every day living. These systems aim to assist older persons in the area of health, safety and comfort. Therefore, they have higher chances to live independent even at higher ages.

1.1.1 Motivation and objectives

One of the main objectives in AAL systems is to maintain the health condition of elderly people. Some criteria are addressed in the following points [2]:

- Nutrition
- Movement
- Mental health
- Social contacts

Ambient Assisted Living systems thus maintain these outlined points which help people to stay healthy and can prevent them from suffering chronic diseases [2]. As a result these persons have higher chances to life independent at their homes.

The highlighted criteria above concentrate on the health condition of a person. Health, safety and comfort are the key areas of AAL systems (see Figure 1.1). All of these three topics are covered by home automation. Intelligent homes should provide devices to determine the health

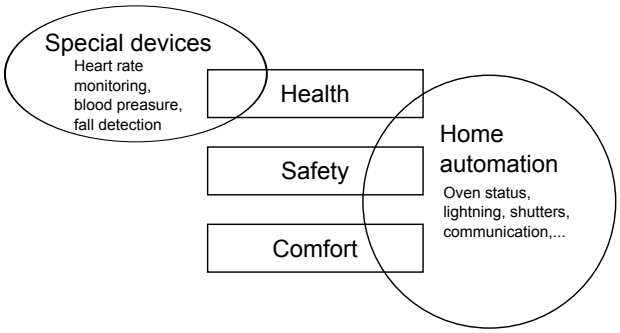


Figure 1.1: Ambient Assisted Living key areas (adapted from [LG07])

condition of a person and its behavior. The observation of the health situation can be done by using body sensors which are also known as wearable electronics. Body sensors can give direct information by e.g. measuring body temperature, blood pressure and heart rate [LG07]. The disadvantage is that the sensors have to be worn all the time. An indirect way to determine the health condition is to equip homes with sensors in the surrounding area. These sensors can be e.g. motion detects, contact sensors at windows and doors and temperature sensors. Therefore, the behavior can be monitored. Based on the activities the health status can be determined. That may be based on the number of activations in a room (e.g. toilet) or from the absence of motion in case of the person falling. When the system identifies a situation as critical, it informs e.g. the doctor, relatives, neighbours or nursing staff automatically for dealing with the situation. The safety aspect ensures that harm is prevented from the person and also from the environment. A critical device in a flat is an oven. An alarm or another action will remind the user to switch off the oven before leaving the flat.

Automatic lighting, phone calls using speed dealing with photos and alarm clocks such as proposed in [3] are used to achieve a better comfort for a person. The comfort functions are mainly achieved by a home automation system.

1.1.2 Current situation and trends

In Europe as well as in other developed regions an enormous change in the demographic structure is evident. In 1920 the average life expectancy was 55 years. In 2010 the average life expectancy increased to over 80 years. Reasons for an older population are, for instance, better medical treatment, a balanced diet, and birth rich periods as was the case in the baby-boomers' period. The baby boom generation (babies born between the years of 1940 and the middle of the 1960s) will rise the population of matured people from 65 to 80 years (increase of about 40 percent) within the years 2010 to 2030 [4].

According to the facts in the second report of the European AAL JP (Joint Program) the figures are slightly different. Their investigations show that the current ratio between the elderly people with an age above 65 years and the working people from age 15 to 64 years was 1:4 in 2008. These numbers will decrease to 1:3 until 2020 and even to 1:2 in 2050 [5].

The business unit for retiree, which is also called the *silver economy* [5], offers new jobs. These jobs are situated in the care sector as well as in the research and development of AAL systems. The demand on health care can be seen in the fact, that we will lack up to 2 million employees in the health and care sector in Europe until the year 2050. The imbalance could be solved or partially solved by AAL systems.

1.1.3 Functional groups

Elderly people can also be classified by their restrictions and how much help they need [2]. Three groups can be identified as following:

Independent living senior citizens ("GO-GOES") have an active life while they are in a good physical and mental condition. Even some deficits in their health condition do not hinder them.

Needy senior citizens ("SLOW-GOES") are facing some limitations in their life which can be because of social issues or health problems. They have to be supported in some activities.

Senior citizens in need for care ("NO-GOES") are in bad physical and mental condition. They always need care at home or at the nursing home.

The primary target group for AAL systems is the group of the *GO-GOES*. The figures for Austria from the year 2004 in [2] show that this group - composing 1.3 million persons - are 70 percent of the persons older than 60. The other 30 percent are formed by the group of the *SLOW-GOES* and *NO-GOES*. For the industry it is more attractive to concentrate on the primary target group because they form the larger number. Additionally, their health condition is better which gives the industry a broader range of products to offer.

1.1.4 Challenges

In the last few sections the current trends were discussed. The faced challenges in AAL systems are briefly presented in the following. Here the major implications are faced. Also possible solutions will be given to overcome or to minimize these shortcomings.

People's willingness

It is a huge challenge to make persons use AAL systems [SDFGB09]. This task is not only addressing the elderly people but also all other participants involved. Professionals and care personal must be considered. In order to have a higher participation it is necessary to understand the consumers' needs and promote such systems. Educational advertising has to be made to overcome some barriers.

In [SDFGB09] the authors describe two main barriers of elderly people that restrain them from using AAL systems. These two issues root either in *psychological* or *technological* issues.

The *psychological* aspect addresses the situation when people are getting older and are falling back into a passive situation rather than being active participating creators. To a designer of an AAL system it is a hard job to provide a proper solution without stressing the passive consumer. But anyway, the role of the elderly people should be an active one which brings a lot of advantages. They can offer their wealth of experience to the younger generations. This way, they will experience that they are still useful for society.

The second restriction lies in the *technical frustration*. At higher age people's willingness to use modern technology is decreasing. Countermeasures can be the early integration of seniors to these products. They should get in touch with such systems even before they need them [SDFGB09]. It is also very important to offer well designed interfaces which are very intuitive and therefore do not need any previous knowledge. The interface design is also related to the topic of usability.

Usability

As mentioned above, the user interface is important for the usability of the AAL system. Some common design paradigms should be followed. A usual approach is to have simple user interfaces. Thus, they should not be overloaded and should only comprise the most important information. Due to the age-induced decreased acuteness of vision the fonts and symbols have to be larger in size.

The usability also depends on the ambient sensors or the body sensors. The ambient sensors should be placed in a subtle way. So the sensors are not distracting the senior citizen. Moreover, they should integrate seamlessly. Body sensors are more invasive. It has to be considered that the sensors are always attached to the body, while in some cases some preparation is needed (e.g. to accomplish good electric contact). It is also possible to integrate most of the sensors in the clothing (wearable electronics).

Fault alarms also reduce the usability of an Ambient Assisted Living system. When having too many wrong alarms the user and the other involved persons will be confused when receiving an alarm. At first they cannot distinguish between a trusted and a false alarm. As a consequence the user will not use the system any more. Furthermore, it can lead to a total shut-down of the whole system.

Social activity

As the assistant technologies grow, less interaction between humans is necessary. As mentioned in the previous section, the *SLOW-GOES* or even the *NO-GOES* have social problems which are effecting the overall health condition in addition. So it is very important that the interactions in these risk groups do not only happen with a computer system. The elderly people should participate in a social community where they can interact with others. One option is to be part of a social network [6]. Social network can e.g. be found within the family, friends and colleges (offline social network).

On the other hand elderly people can use online social networks. These networks are rather used to maintain relationships with other persons than to create new friendships. Additionally to well known social network, some custom networks for particularly elderly persons are available.

Online social networks can provide status information, photo exchange, gaming as well as some other functions [6]. Gaming helps to keep a healthy psychological condition and also maintains the social interactions between the users.

There are also some efforts to bind the health status to the profile. Therefore, an interface between the health monitoring system and the social network has to be included. The health condition is just visible to a group of persons (e.g. care personal, relatives, and doctors) that can take advantage of the additional information.

Other issues

In order to evaluate the needs of the target group (*GO-GOES*) some specialists have to work together. According to [2] the best results are met when interdisciplinary teams are formed, including e.g. social scientist, psychologists and ethicists.

Another challenge that has not been addressed yet is that the AAL designers are usually younger than the users. So they are not able to put themselves into the position of the elderly persons. Moreover, the designers are not able to meet the needs of the seniors with their implementation. To tackle this problem the elderly persons should participate in the development process. This makes it easier to develop the appropriate products. The acceptance will also increase.

1.2 ATTEND project

The project ATTEND (AdapTive scenario recogniTion for Emergency and Need Detection) was founded in 2009. The involved organisations are the Vienna University of Technology, Institute of Computer Technology and the sub contractor CEIT REALTEC (Central European Institute of Technology - Institute for Rehabilitation and Assisted Living Technologies) [7].

The goal of this project is the investigation and prototyping of a system which supports elderly people in an extended autonomous living. The infrastructure is a sensor network enabling the analysis and detection of abnormal behavior. When an unexpected behavior is recognized, an alarm is raised. These alarms are handled by the person itself, neighbours or nursing staff.

The sensors should integrate seamlessly and should ensure the person's privacy. Consequential, cameras and microphones are not used. Body sensors and control elements on the sensor nodes are also not part of the system. Great focus lies on minimizing the installation and maintenance work.

The contribution of CEIT REALTEC is the development of the hardware. The hardware should be installed in an environment. Therefore, the installed sensors should record the data of a person as well as other environmental data (e.g. temperature). The collected data should be submitted to the Vienna University of Technology, Institute of Computer Technology in order to analyse and process the data.

The Institute of Computer Technology works on analysing and recognizing a person's activity and behavior. The key points that should enable to fulfil this tasks are the development of machine learning algorithms, Sensor Fusion and symbolic data processing [YB09]. The challenges describe as follows:

1. Machine learning: development of reliable and fast methods - fitted to the specific sensor type, groups of sensors or on semantic symbols - modelling daily routines.

2. Situation and scenario recognition: detecting the most important situation in the routines of the assisted person.

3. Sensor Fusion: investigation of the additional benefits of combining redundant sensors and different sensors types in order to model a person's behavior. The more complete knowledge should be gained from only a few sensors.

4. Combination of rule-based and learning systems: A main aspect is to combine the proposed methods. This should be done to make the system more robust in terms of alternating behavior of the person(s). The combination of machine learning and Sensor Fusion should take place.

The recent work of Yin and Bruckner has solved the first two topics, addressed above. In [YB10b] the machine learning approach is highlighted. The authors propose a way to describe the temporal distribution of measurements of a specific sensor. Aspects of the second task are described in [YB10a], where a person's behavior is modelled by using hidden Markov models.

1.3 Putting this work into context

The input to the ATTEND project from this thesis is the aspect of Sensor Fusion. Since the expected data are not without disturbances the combination of redundant sensor information as well as information from various sources are used to obtain a more complete picture. Sensor Fusion thus delivers fault tolerant behavior which is described in Chapter 4 in more detail.

General description

The main goal of this thesis is to develop and adapt Sensor Fusion algorithms which apply to Ambient Assisted Living systems. The aim is to get more significant data in order to describe the behavior of a person. The behavior recognition process is not part of this thesis, though.

The measurement data is provided by the cooperation partner CEIT RALTEC. The company made three datasets available which contain various data from different sensor and types. The introduced algorithms should be implemented using MATLAB.

The tasks can be divided into two subtasks. The first one should implement the fusion algorithms on the raw data level. The second assignment is to reduce the configuration work of an Ambient Assisted Living system.

After the fusion process the data is supposed to be more reliable so that less incorrect alarms are raised by such a system. The added fault tolerance by Sensor Fusion should reduce incorrect alarms and thus improve usability.

Combination of raw data

At the raw data level a set of possible algorithms should be evaluated in order to give a comparison and pick out the most appropriate of all possibilities. The algorithms should draw as much information from various sources as possible. This should have the effect that the combined information is more unambiguous and thus identifies a situation more clearly. The picked algorithm should tolerate sensor faults and should also lower uncertainties.

Increasing usability

Usability is another issue that has to be solved in terms of configuration work and fault tolerance. The task is thus to have as little configuration work as possible at the installations phase.

Therefore, all included sensor types of a node have to be identified automatically by their sensor readings. After this process the data of the movement sensors should be used to get a representation of the sensor arrangements without knowing the real topology. The hardest task will be the identification of faulty movement data without any other knowledge. These incorrect movement data should be determined by using the topology. The improved position information should be the fundament for further works in the study of a person's behavior.

2 Environment

The data for the thesis was taken from the ATTEND project. The sub contractor CEIT RALTEC provides three data sets. Ambient Assisted Living environments were introduced to three test persons (TP2, TP3 and TP4). These test persons lived in these assisted environments for about four months. During this time the installed sensor nodes recorded measurements. These measurements were sent to a central unit which stored them into a database. A sensor node can be equipped with different types of sensors. The used sensor types are reed sensors, accelerometers (measuring vibrations), temperature sensors, light sensors, and PIR (passive infra-red) sensors. The sensor node configuration depends on the place of installation and the intended purpose.

The description of each flat with the installed sensor nodes has been provided. The test person had to answer the questions of a form. The documentation was done on each day of the observation period. So that the gained records illustrate the ways in which the individual test person lives. These documents are the diaries of a person which include documentation records concerning the health condition, sleeping time, cooking behavior, visitors, and interaction with the system.

The next Section provides a description of a flat where the data was recorded in. It shows how sensor placement was done and which properties were measured with each node. The subsequent sections show the capabilities of each sensor type and the data format in which the data is provided.

2.1 Representative flat with sensor placement

As a good representation of all given flats, the environment of test person 2 is given in Figure 2.1. The corresponding Table 2.1 shows the sensor configuration of each sensor node. In the graphic a sensor node is depicted as an orange rectangle. The node has a blue frame if it contains a reed contact. An included movement sensor is indicated by an arrow which shows the direction of observation. The interface unit of the system is listed in the figures as LUI (Local User Interface). The central station is here called the HCU (e-Home Central Unit). The WIFI (Wireless Fidelity) symbol indicates a wireless modem.

Figure 2.1 depicts the flat of test person 2. It comprises eight rooms where two of them are not equipped with movement sensors and one is not used at all, containing no nodes. The entrance door is next to sensor node 141. In addition to the given figure, Table 2.1 contains all sensor nodes with their placements and the including the sensor types. The sensor node under the bed, for example, is just equipped with an accelerometer, temperature sensor, and a light sensor. The

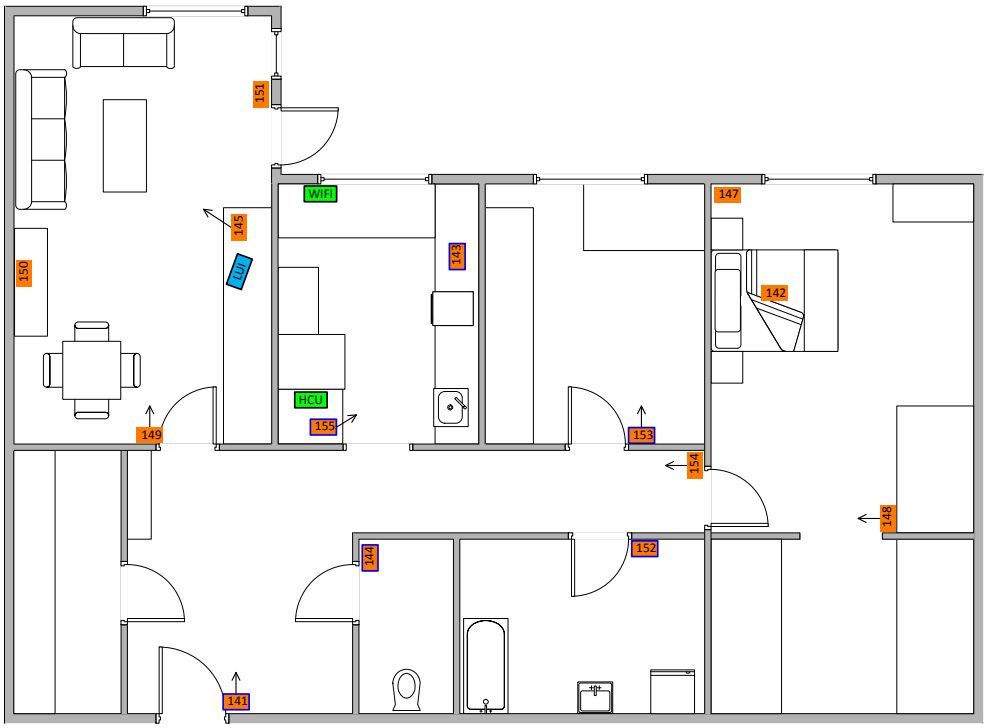


Figure 2.1: Flat of test person 2

Table 2.1: Sensor configuration in flat 2

sensor node	placement	reed	accel.	temp.	PIR	light
141	entrance door	✓		✓	✓	✓
142	bed		✓	✓		✓
143	oven	✓		✓		✓
144	rest room	✓		✓		✓
145	living room	✓		✓	✓	✓
147	next to the bed		✓	✓		✓
148	bedroom floor		✓	✓	✓	✓
149	living room entrance		✓	✓	✓	✓
150	under commode living room		✓	✓		✓
151	next to entrance door		✓	✓		✓
152	bathroom	✓		✓		✓
153	home office	✓		✓	✓	✓
154	corridor	✓	✓	✓	✓	✓
155	fridge	✓		✓	✓	✓

purpose of this sensor is to determine the presence of a person. Sensor node 154, located next to the bedroom door, includes all used types so that it is also able to detect movements and door activations.

2.2 Sensor Description

The used sensor types are discussed in this Section. In order to describe the communication behavior for each type the sending intervals are given. The maximum sending interval determines the time between two transmitted sensor values without significant change of the measured property. The maximum sending interval applies to the temperature sensors and the light sensors.

Reed contact

The reed contact is used to detect an open or a closed door. At the opening event the sensor offers a "1" and at the closing event it shows a "0". The minimum sending time between two events is 100 ms.

Accelerometer

The accelerometer was used to measure the degree of tremor on the floor. The minimal sending interval is 200 ms which is equal to the minimum duration of a vibration. The sensitivity is 40 milli g, while g is the gravity acceleration.

Temperature sensor

The temperature is measured in degree centigrade. This sensor reacts when the value is changing more than ±0.5℃. In that case the minimum sending interval is one minute. If the value stays within the range, the next value will be send after one hour (maximum sending interval).

PIR sensor

The minimum sending interval is 3 seconds. The sensor is sending a "1" if movement is detected in its range. After the movement the sensor is blocked so that it cannot send any other movement information for 3 seconds. The only value a movement sensor is able to send is a "1". The sensor has an approximate detection range of 6 meters with an opening angle of 140°.

Light sensor

The maximal sending interval is 10 minutes while the minimum is 1 minute. The measurement is given in lux, while a new measurement is transmitted if the intensity changed ±10% or ±100 lux.

2.3 Data format

The data sets are stored in cvs (comma-separated values) files. Each row represents one measurement, while the individual columns have a special meaning. The meaning of each column is given in the following order:

- ID - primary key

- Datetime - timestamp in the format YYYY-MM-DD HH:MM:SS
- ms - extention of the date giving the milliseconds
- NodeID - number of the sensor node
- SubType - indicates the sensor included in the sensor node
  - 1 - reed contact
  - 2 - accelerometer
  - 3 - temperature sensor
  - 5 - PIR sensor
  - 6 - light sensor
- Val0 - sensor value

By knowing all these details, the data sets can be read and interpreted correctly.

3 State of the Art

The introduction has made the reader familiar with the basic concepts of Ambient Assisted Living systems and the ATTEND project. In order to get some ideas for the implementation, the current work in this field has to be discussed.

The first Section introduces the reader to the theoretical background of Ambient Assisted Living. Especially, the situation in Europe and the project initiatives are reflected. Section 3.2 deals with the current work, combining Sensor Fusion and Ambient Assisted Living. Since these works are not fulfilling the requirements of this thesis, related implementations from different fields will be highlighted in the final Section.

A separate chapter is devoted to the field of Sensors Fusion (see Chapter 4) since it is the main aspect of this thesis.

3.1 Ambient Assisted Living

One of the main research and development programs was introduced by the European Union [8]. The program will be addressed in the next section, dealing with projects and financial funding in Europe. A general architecture is presented in the following section. The architecture is appropriate for a large range of Ambient Assisted Living implementations. The final part provides information about current projects.

3.1.1 Efforts in the European Union

The European Union has introduced its own program to boost the research and development (R&D) of Ambient Assisted Living. The program is called Ambient Assisted Living Joint Program (AAL JP) [8]. The project was first introduced in 2008, including 20 member states (e.g. Austria, Belgium, Germany, Finland, France, Poland, and Sweden) and three associated countries (Switzerland, Norway, and Israel) [9]. Figure 3.1 shows the European map with the number of organisations (958 organisations in total, covering 117 projects) highlighted [10]. The colour indicates the number of participating organisations in each country. Compared to other European countries, Austria has a middle ranking in the number of participating partners.

In [10] and [4] an overview of the participating organisations is given. The data are derived from the first AAL JP call from 2008 which are presented in Figure 3.2. The largest number is devoted

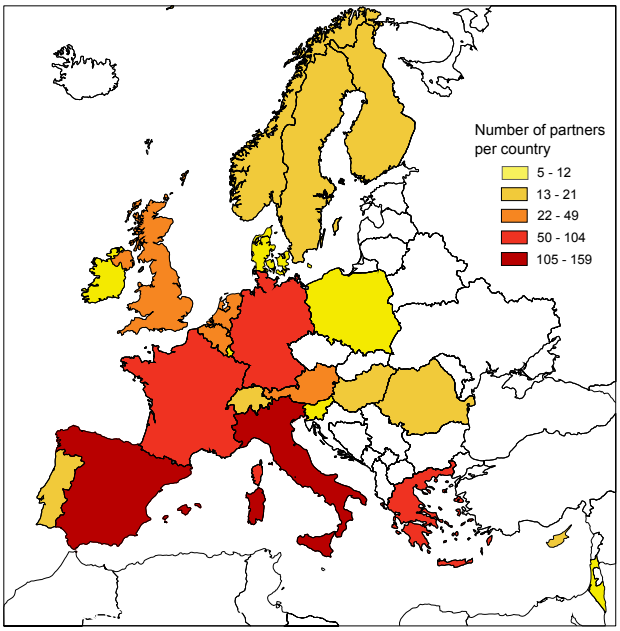


Figure 3.1: AAL JP participating countries with number of partners respectively [10]

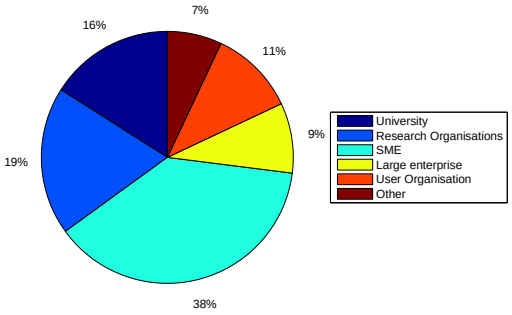


Figure 3.2: Participating organisations in the AAL JP [10] (SME. . . Small and Medium Sized Enterprise)

to the small and medium sized enterprises (SME) sector with a total of 38 percent followed by research organisations (19 percent) and universities (16 percent). The funding was decided in accordance with article 185 of the Treaty on the Functioning of the European Union (TFEU) [5]. The AAL JP will have a total duration of six years, starting from 2008 until 2013. €600 million will be the total investment [Vod08] where 50 percent will be from public capital and the other half will be from private investments. According to the first call to the AAL project [4], Table 3.1 shows the funding contribution of the AAL JP partners. The main goals of the AAL JP are the creation of information and communication technology (ICT)-based products and services, the stimulation of development, research and innovation work [5], and the improvement of the industrial usage of the gained knowledge from these projects. All projects last two to three years and should be ready for the market after this time.

Finally, some projects are highlighted which are related to the ATTEND project. *CARE – Safe Private Homes for Elderly Persons* is funded with €1.7 million. The realisation implements

Table 3.1: Funding contribution of the participating countries

AAL JP partner state	Call 1 (2008) (million €)
Austria	2.5
Belgium	1.0
Germany	5.0
Finland	2.5
France	2.5
Israel	0.5
Norway	1.0
Poland	0.5
Sweden	0.8
Switzerland	2.0

an alarm and monitoring system for elderly peoples' homes [11]. The system may spot falls and will automatically alarm the person responsible. The project has a total duration of 30 months and started in July 2009. *REMOTE* is a project that deals with the development of a software architecture to improve an Ambient Assisted Living environment by applying health monitoring and behavior recognition techniques. The participating countries Germany, Italy, Iceland, Norway, Spain will have a total budget of €2.2 million [10]. The last project which will be discussed is *SOFTCARE − Kit for Elderly Behaviour Monitoring by Localisation Recognition and Remote Sensoring* with a total funding of €0.7 million. The home monitoring system consists of ZIGBEE sensor nodes [12]. They are used to find behavior patterns in a person's daily activities. The system is non-invasive because no cameras and body sensors are used.

There are lots of other projects such as the *ALADDIN*, *AMICA*, *DOMEO*, *Happy Ageing* and *HELP*. A brief description and an overview of the funding can be found in [10].

3.1.2 Generalized Ambient Assisted Living architecture

The following details will provide the reader with information about structure of an Ambient Assisted Living system. In this context the possibilities of behavior monitoring and alarming is given. Figure 3.3 portrays information flow between the user's home (smart house) to a health service provider and to other involved participants. The health status is recorded at the smart house. In Chapter 1 the two options of obtaining the health status have already been mentioned. The information can either be gained from body sensors or from sensors in the environment. The actual data are presented on a local user interface or on a mobile device. The local user interface provides more services such as video telephony, reminder functions, and alarm functions. The information transmitted to the coordination centre is stored at the health care provider so that the trends with regard to the health condition can also be reviewed. The records can be used for personal purpose or for having supervised treatment by an expert. In [MCM+11] the visualisation of the health data is done via a web interface.

Also the commercial aspect is considered. These systems may also include further services to order products such as food, medicine, and other required things.

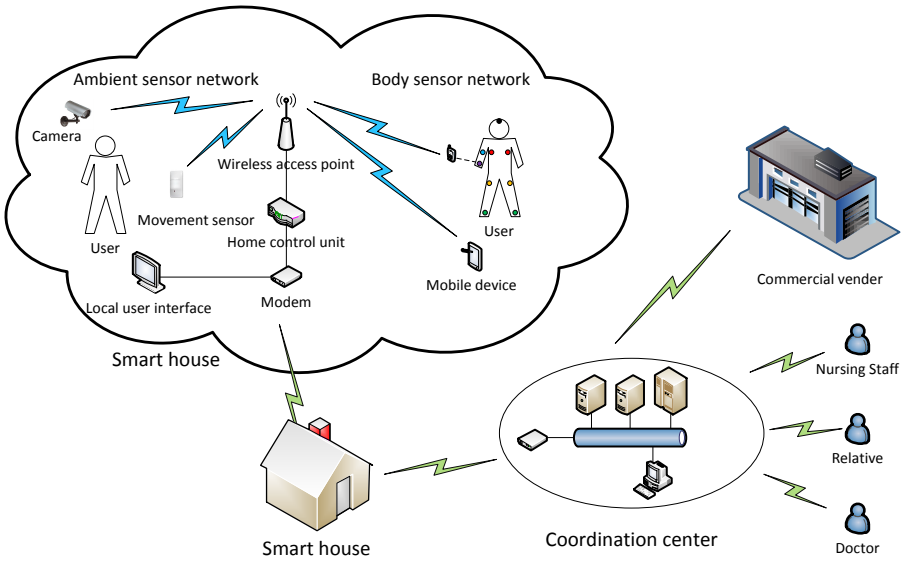


Figure 3.3: Generalized architecture of an Ambient Assisted Living system (adapted from [SDFGB08])

In the thesis the focus is on the data processing which is all done within the assisted environment. So the other services outside the local environment are not discussed. The only aspect of the outside world is the notification of experts. When the used algorithms reject incorrect measurements, the involved persons are not bothered with incorrect alarms.

3.1.3 Data monitoring

The monitoring system forms the part of an Ambient Assisted Living system in which all pieces of information are collected and processed. As already mentioned the data can be collected with ambient sensors and body sensors.

Body sensors

There are a lot of ways to monitor vital signs. Figure 3.4 depicts some types of sensors which can be attached to the body. Starting from the head - the electroencephalogram (EEG) is used to determine the activity of the brain. Therefore, electrodes are attached to the head. The blood pressure can be measured with non-invasive methods as proposed in [Kha03]. The pulse oximetry is used to determine the oxygen concentration in the blood. Therefore, a sensor is placed on the fingertip or the ear which transmits light impulses through the skin which are measured later on. The amplitude of the transmitted light impulse depends on the oxygen saturation of the blood and thus gives evidence to this bio-signal [CGV+11]. The electrocardiogram gives the heart rate. Furthermore, it can be used to determine heard-related diseases. The method of electromyography (EMG) is used to determine muscular disorders. The pose and activity of a proband are estimated via motion sensors such as accelerometers and gyroscopes.

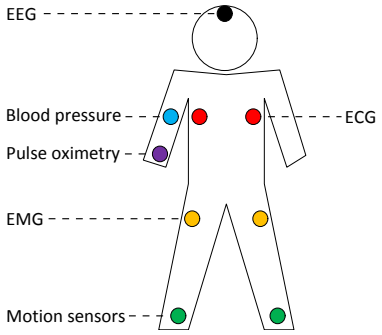


Figure 3.4: Body sensors (ECG ... Electrocardiogram, EEG ... Electroencephalogram, EMG ... Electromyography) (adapted from [CGV+11])

Ambient sensors

The ambient of an assisted home is monitored by many different sensors. There are sensors which observe environmental conditions such as temperature, humidity, and light conditions. However, there are also sensors that determine the person's activity. Therefore, the most prominent types are movement sensors, accelerometers, and reed contacts. These sensor types can also be found in the used environment which is given in Chapter 2. Pressure mats as in [Mit08] have a better resolution than the other types like accelerometers and movements sensors. Video cameras and microphones are also used in this application domain [TJDS09]. The drawback of this kind of observation is the acceptance of the user. The user might be intimidated when someone is able to watch him/her at his/her home. Another possibility is also to equip everyday items like kettles with sensors [BGP+10]. The sensor gives information about the filling level and temperature as well as the power status and the tilt information. The included sensors are used to determine the behavior which is described in the next Section.

3.1.4 Behavior analysis

The used method for behavior analysis in the ATTEND project are hidden Markov models [YB10a]. A statistical approach is applied in these models. A hidden Markov model can be considered a finite state machine, including state transition probabilities and output probabilities [Bru07]. The term "hidden" indicates that the states cannot be seen from the outside. Only the output variables are visible.

The probabilities of the model are gained from a learning process. When observing a sequence of outputs, the sequence of states producing these outputs must be found. Therefore, the forward- and Viterbi algorithms are used to find the sequence and to calculate the probability of this sequence [YB10a]. If the observed sequence does not fit the Markov model, the probability will be low, thus indicating unusual behavior.

Finding the longest common sequence is another method of behavior analysis [PBV+09]. In doing so, a reference sequence is compared to an actual sequence. A set of reference sequences - a dictionary - is gained from previous observations. When similarities to the dictionary are present, the situation is classified as normal. Otherwise unusual behavior has occurred.

3.2 Sensor Fusion in Ambient Assisted Living

A lot of Ambient Assisted Living projects implement Sensor Fusion techniques which use body sensors as information sources. In [ZTG+10] different bio-signals are used to determine the level of stress of a proband. [LT08] uses a multivariate autoregressive (MAR) model as data-fuser for ECG related bio-signals. Erroneous sensors do not influence the operation. Alarms are also sent when a critical health condition is monitored. Due to the combination of the diverse sensors, situations with increased physical activity can be determined so that these conditions don't cause incorrect alarms.

Fall detection implementations are found in [13] by tracking the person's head and by computing the vertical and horizontal velocity. Another approach is presented in [DNWZ08]: the authors propose to identify a potential fall via accelerometers. These accelerometers are ambient sensors built into sensor nodes, observing the floor's vibrations. Therefore, no attachment to the body is necessary.

[ZS09] uses two accelerometers: one is attached to the waist and one to the foot. Coarse classifications are done to extract features of each sensor. Two neural networks categorize the data into three states. The gained states of the neuronal networks are later fused into a common representation. Later on a heuristic algorithm is used to get the transitions between the activities e.g. from sitting to standing. After the course classification a hidden Markov model is used to recognize activities of a time series.

The gap between behavior analysis and health conditioning is closed by the work of [PBU10]. In their work they propose to combine activity data (gained from accelerometers) and the emotional status of a person (gained from the physiological sensors). Thus, they are able to distinguish between activities like running, walking, and cycling. The mental component has already been introduced. In relation the mental component, mental health and social contacts have already been discussed in the Sections 1.1.1 and 1.1.4.

Behavior extraction is also done in [BHG+09]. Only simple hardware such as wireless webcams and the sensors of a mobile phone is utilized. Therefore, the data of the mechanical features and the visual data are fused into one representation. The included sensors of a mobile phone deliver some features. Features of walking are, for example, gained from the analysis of the Fourier transformation of the accelerometer data. The features of a accelerometer cannot classify complex actions like brushing one's teeth. The spatial information from the video data is needed to identify these activities. When having various video sources, the observations can be combined to a three-dimensional representation. The camera's information indicates whether a person, for example, has fallen.

A similar hardware configuration in [RHS10] is used for another purpose. Therefore, they use infrared cameras and the orientation data from a mobile phone to control devices in their surrounding. The cameras are used to determine the position of the person in the room. Furthermore, the orientation of the cell phone is used to discover manageable devices. These devices are displayed on the mobile phone. The application is used to generate a better usability by using cameras and the phone only.

In order to deal with more persons in an environment via cameras the discrimination between the persons is needed. In [TJDS09] a person is equipped with a wireless accelerometer. Every sensor has a unique identification number which is associated with the person later on. The movement

of the video data and the accelerometers are compared in order to find the best match. The person in the scene can thus be identified by the accelerometer identification number.

A non-invasive monitoring system is used in [NAQB+10]. The idea is turning the electrical appliances into sensors. The proposed system is continuously monitoring the power consumption of each device. When switching off or on a device, characteristic voltage spikes occur. This feature is used to discriminate between the individual appliances. After an initial learning interval the electrical properties of the devices are gathered. The activities are determined by the switched on/off appliances. The fusion process is based on Bayesian statistics. The device activations are considered according to the probability associated with an activity. Other factors such as time of the day, number of activations, and external conditions are also part of the decision process.

Sensors Fusion is included in all presented methods but not all of the systems are based on ambient sensors. There are also many implementations using cameras, which may restrict peoples' privacy. The presented implementations do only fit partially to the assigned tasks of this thesis. Therefore, other related applications have to be considered, which will be presented in the next Section.

3.3 Related work

Related fields and applications are introduced to the reader. The presented concepts are adapted later on and used in the implementation. The first two methods deal with fault tolerance in a sensor network and the other applications handle the interactions between sensor nodes and their representation. In a later step, Chapter 5 will give the modifications to fit the requirements of the assigned tasks.

The work in [DLC05] concentrates on the influence of faulty sensor measurements. They deal with sensor faults in many different ways. In order to find a "correct" sensor, the most promising experiment was done by using the linear randomized voting algorithm. The idea is to select a random sensor and collect votes, confirming the measurement of this sensor. In comparison with an equivalent sorting algorithm, a better performance is observed. The algorithms show the same performance at small sized sensor networks but at a higher number of sensors the linear randomized voting algorithm has better performance. As a result of this process one alleged correct sensor measurement is found. The sensor's variance expresses the uncertainty. When the fused output should have lower uncertainty, at least two or more sensors have to be considered. For this thesis it is recommended to evaluate the algorithm several times so that the data from more sensors is available.

The others of [SQSX10] also deal with fault tolerance. The concept of fusing raw data is extended: the impact of a sensor measurement is determined by the uncertainty (variance) of a sensor. The variances are set initially and are re-calculated in every iteration step. After each progressing steps the variances do not reflect the uncertainty, given in the sensors' data sheets. Moreover, they reflect the deviation of the measurement from the fused result. One can observe that the gained benefit is that the configuration work, concerning sensor parameters, need not be done.

Target tracking is also a very prominent application domain. Usually, modern navigation systems use the benefits of Sensor Fusion. Such systems are found in [RM04] and [14]. The most dominant fusion approaches for such applications are Kalman filters. These filters incorporate a model of the environment. In the first step a Kalman filter predicts the next model output according to the most recent values. In a second step the estimation is corrected by the intake of a new

observation. The method offers more accuracy than one without fusion. The fusion algorithms do not only use data from the global positioning system (GPS) but also from other location data. [RM04] also used the mobile phone location services (MPLS) as an additional data source but other sources like radar and sonar could also be used. The context to Ambient Assisted Living is not clear in the first moment because the positioning techniques do not work inside a building. The explained methods, however can be applied by the nursing staff. When an alarm goes off, the next task is to inform the appropriate person. So the health provider could determine the position of the closest person next to the assisted one. It also works when the person is out of reach of the GPS signal. The mobile phone location service could give rough location information. All the outlined points are not concentrated on the main aspect - tracking the person inside a living environment.

The basic idea has to be adapted to binary sensors as they are used in [ZW10]. The implementation uses the Kalman filtering but other researchers like [OE07] and [SG06] solve this problem with optimization algorithms. The next evolution of such algorithms is that they are able to track multiple targets simultaneously - [DSK+07], [AK08] and [ZC04]. The "premium" version is presented in [ILC09] and [COM+06] where in addition to the position, the velocity and acceleration are part of the output. All presented methods have the assumption of required overlapping observation areas in common. In an assisted environment this condition cannot be considered to be present. In the given sensor placement in Chapter 2 it is evident that the installed sensors do not cover all areas of the room and do not have overlapping observation areas in all cases. Therefore, [ZC04] proposed an algorithm for re-positioning sensors. However, this does not solve the fundamental problem that the geometrical conditions have to be known either.

Therefore, another approach from [GWET05] has to be considered. The authors present an error reduction algorithm for sensor networks. The method divides an area of interest into smaller areas. These areas are defined by the included sensors. Sensors can be assigned to more than one area. The author's terminology and representation of the areas is displayed in Figure 3.5. The individual regions R_i $(i = 1, 2, \ldots, n)$ form the entries of the binary matrix in which n is

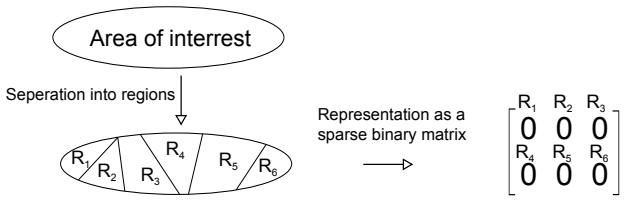


Figure 3.5: Matrix representation of the mapped areas (adapted from [GWET05])

the number of regions. The depicted case shows that none of the areas is occupied - no event was detected. The fused data that is associated with an area is stored and when an event occurs, this data is compared to the new one. If a significant change happens, the representative cell is set to 1, otherwise to 0. For error reduction a probability matrix, having the same size as the binary matrix, is introduced. The entries are mapped in the same way. The matrix represents the probabilities of getting from one region into another one. When activations happen the current position is used to determine how likely the current event is. One drawback of this method is that the position of the sensors has to be known approximately for mapping them into the areas. A premise of this thesis is that as little configuration work as possible should be done. Since mapping sensors to different areas, the method does not support this criterion initially. It could be adapted so that every region includes only one sensor. The authors also mentioned that the

tracking error increases when more targets are present. The intention of the tracking algorithm is just to track one person because the behavior of this specific person is under observation. The behavior extraction should end when more persons are present. So this situation has to be determined.

A similar method is presented in [PCH08]. The authors build a global view of a sensor network, using the local information of the individual sensors. The application domain are surveillance systems in which the sensing devices are cameras and microphones. The gained information from the local nodes is used to build a global representation. The process takes less configuration effort for the user. It doesn't even take a central processing unit to combine the local features. The global representation is built by using the correlations among individual sensor information. These correlations are learnt by applying the Boltzmann machine learning approach. Three phases are addressed for the overall application. The first phase is the learning phase in which features are extracted from the audio- and video data. The second phase presents the network establishment when the correlations among the nodes are found. In the last phase unusual events are detected. Therefore, the neighbour nodes send additional information to overcome uncertainties. In the network establishment phase the correlations are computed by setting up the compatibility matrix among pairwise nodes. A node is represented as a state vector: every state can take the logical values true or false. When observing pairwise sensors, all these combinations have to be considered. This method creates a global view of the entire network. The global view is similar to the probability matrix of [GWET05]. The algorithm, applied to Ambient Assisted Living systems, offers the relations between the sensors but the representation for the valid activation has to be searched for.

The missing link is found in the work of Mitterbauer. In [Mit08] - trying to model the behavior of a person - he presents the concept of the person model. In the environment of movement sensors and pressure mats, the person model was used to distinguish among individual persons in the room. The person model includes the activated sensors. Thus, the person model gives the current position.

4 Sensor Fusion

Since Sensor Fusion is the main topic of this thesis, a separate chapter is devoted to this topic. The basic idea of Sensor Fusion, including the drawbacks of conventional sensor systems is given. It is also shown that some limitations have to be faced.

The first sections make the reader familiar with the concept. Section 4.2 presents different ways to classify the fusion process based on selected criteria. The next section introduces the reader to the basic architectures and gives the advantages and drawbacks of the individual architectures. The final section is dedicated to fusion models. These fusion models describe the fundamental steps in the fusion process. Depending on the model, these steps are broken down in more detail.

4.1 Concept

Sensor Fusion has already been used in nature before it was adapted in a technical context. Animals as well as humans are able to combine the information of their sensing organs to generate a better understanding of the overall situation. In many situations it is necessary to combine the different senses. For example, a liquid is supposed to be water but the smell or even taste tells that it is harmful. An animal could also take advantage of the olfactory organ and aural sense to determine if there is a predator and therefore could get to a safe place.

In a technical environment several definitions in respect to the origin of the data are offered. The definition of Sensor Fusion is given in the next Section.

4.1.1 Definitions

There are some misconceptions about the terminology that is used in the different fusion concepts. In literature the terms of data fusion, information fusion, multi-sensor-data fusion and Sensor Fusion are used. Some of the expressions are used in the wrong context. But usually the fusion concept refers to the origin of the data that is fused.

The authors of [SBW99] define data fusion as *"the process of combining data to refine state estimates and predictions"*. The data can be of various sources (e.g. database, knowledge, ...). When receiving the data from sensors, generally the term *Sensor Fusion* is used. Hall and Llinas used *multisensor data fusion* in context of fusing data from various sensors [HL97].

In order to avoid an ambiguous interpretation the similar term *information fusion* can be used instead of *data fusion*. In [Rao10, p.xxvii] the following definition of information fusion can be found:

> "Information fusion encompasses the theory, methods, and tools conceived and used for exploiting synergy in information acquired from multiple sources (sensors, databases, information gathered by human senses, etc.). The resulting final understanding of the object (or a process or scene), decision, or action is in some sense better, qualitatively or quantitatively, in terms of accuracy, robustness, etc., and more intelligent than would be possible if any of these sources were used individually without such synergy exploitation."

Moreover, we can define Sensor Fusion as the combination of sensor data or data produced from sensory data to a more complete information than it would be possible from an individual sensor. In this context the terms of Sensor Fusion and information fusion will be used. Data fusion can be mistaken with *raw-data fusion* (see Section 4.2.2).

Sensor Fusion does not assume that the sensors are of identical type. Information has to be brought to common context. Therefore, either a model or a technique of sensor value normalization like in [Mit07, p.97ff] can be used. The methods of sensor normalization convert different pieces of information to a common scale. The common applied techniques are binarization, parametric normalization functions, and conversion to probabilities.

4.1.2 Sensor limitations

The definition of Sensor Fusion suggests as potential reason that single sensor systems that do not perform any kind of fusing do have to face some disadvantages. Elmenreich provides the following list [Elm02, p.9f]:

Sensor deprivation: The failure of a sensor causes the loss of information, so that the desired object cannot be observed sufficiently or completely.

Limited spatial coverage: Every sensor is considered to be limited in the range of observation. If we consider a movement sensor, it faces a limited coverage angle and depth. Potential shading due to the geometric conditions must also be considered.

Limited temporal coverage: A lot of sensors cannot take continued measurements. One reason for this could be the set-up time of a sensor which it takes a sensor to provide an accurate result. Analog to digital converters also impair this kind of delay from applying a changed input to a stable output. In digital applications the sampling rate limits the temporal resolution.

Imprecision: A single sensor is limited in its precision which is defined by its ability to resolve small changes of the measured variable.

Uncertainty: In contrast to imprecision the uncertainty arises from the object rather than from the sensor. Uncertainty will be seen when properties or features cannot be completely observed or if the results are ambiguous [Mur96].

4.1.3 Benefits of Sensor Fusion

Sensor Fusion aims to improve or even overcome the shortcomings of a single sensor. The following points will illustrate the benefits of Sensor Fusion [BRG96]:

Robustness and reliability

Extended spatial coverage

Extended temporal coverage

Increased confidence

Reduced ambiguity and uncertainty: More information of an object reduces the chances to associate wrong or uncertain relations about a situation.

Robustness against interference: By applying different measurement methods the overall system is less sensitive against interference. If we consider measuring the revolutions per minute of a wheel, using different types of sensors, it is possible to do that with optical or magnetic sensors. While optical sensors are sensitive to pollution, magnetic sensors are to magnetic interference. The optimum would be the combination both of them. In case of interference the best suitable sensor has to be chosen.

Improved resolution: By combining the measures of multiple sensors the resolution can be improved.

4.1.4 Catastrophic fusion and limitations

Basically, *catastrophic fusion* occurs when the efficiency of a multi-sensor fusion system is much lower than the operational performance of one or more individual sensors [Mit07].

A sensor is designed to work correctly under specific environmental conditions. If a sensor is facing some conditions in which it is not supposed to have accurate results, this single sensor is generating a wrong output, leading to an overall catastrophic fusion. In order to solve this problem secondary classifiers have to be introduced. They offer a classification of the single results of each sensor. Therefore, bad performing sensors can be neglected. Algorithms (see Chapter 5) like the confidence weighted averaging fusion will perform so by assigning the weight of zero to an incorrect sensor.

Movellan and Mineiro [Mov98] see the issues of catastrophic fusion when single components outperform the overall system after the fusion process. Lucey, Sridharan and Chandran are also facing the same problem in [LSC01], performing the fusion of audio and video data. Another restriction to Sensor Fusion is that it can not turn “bad” data into “good” one. Concerning the appropriate fusion algorithm it is vital to note that additional information that is provided by the input set cannot be extracted to get a satisfying output. For instance Sensor Fusion cannot guarantee to reconstruct the meaning of a noisy or distorted input signal when the signal-to-noise ratio (SNR) is to low. In [BI96] Brooks and Iyengar show that even the iteration of a Sensor Fusion algorithm does not improve the precision or accuracy of the fused output.

4.2 Classification based on the types of fusion

The fusion process can be classified according to several properties. These may be the sensor arrangement, input/output type of the data or level of fusion. The following sections will discuss these classifications.

4.2.1 Sensor configuration

We can distinguish between three different types of fusion based on the sensor configuration - *complementary*, *competitive*, and *cooperative fusion* as visualized in Figure 4.1.

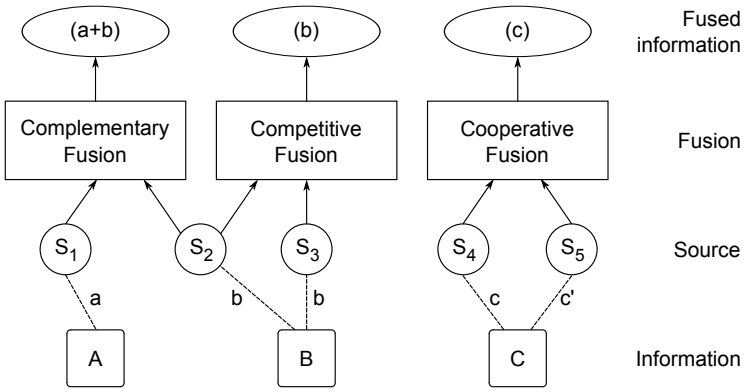


Figure 4.1: Complementary, Competitive, and Cooperative Fusion (adapted from [Elm02])

Complementary Fusion: The model fuses some partial information from different sources to obtain a more complete representation. The single sources are not overlapping. According to Figure 4.1, S_1 and S_2 combine different pieces of information (a and b) to the fused information (a+b). A real-life example is surveillance cameras: each camera just covers a certain area of view. In the combination with the other cameras a more complete description of the area of interest is found. Another example could be two temperature measurements taken at two different locations.

Competitive Fusion: The information from independent sources providing redundant data can be fused to improve the confidence level of the overall output [NLF07, p.7]. In Figure 4.1 S_2 and S_3 provide the same information b about B. This sensor configuration is reliable and thus shows fault-tolerant behavior. Different algorithms can be applied to this type of fusion to detect erroneous sensors [DLC05]. In case of a faulty sensor the information can be used from the remaining sources.

Cooperative Fusion: The term cooperative refers to the process of fusing data from one or more sources to get new information that would not have been possible to obtain from a single source. As shown in Figure 4.1, S_4 and S_5 provide the information c and c' to fuse it into new information (c). An example could be the fusion of two pictures from a stereo camera to obtain a three dimensional image of a scene.

The above mentioned types do not exclude each other. Further combinations can be used. According to the hierarchical architecture (see Section 4.3.3) the individual fusion types can be applied at one level. Furthermore, one of the three mentioned types of fusion can be used at a higher level.

4.2.2 Levels of fusion

Three levels of fusion can be distinguished - *raw data fusion*, *feature fusion*, and *decision fusion*. The others of [NLF07, p.8] also referred to them as low-level fusion, intermediate and high-level fusion, respectively.

Raw Data Fusion represents the fusion of unprocessed data from a number of sources to a more accurate (e.g. less noisy) representation. Therefore, each source needs to observe the same property of an object.

Feature Fusion combines features that have been derived from different sensors. The single feature takes less communication effort than the various features or even the raw sensor data. The transition from a set of features to a single feature suggests that some kind of information loss is obvious.

Decision Fusion obtains decisions from various sources and fuses them into a final decision. Applied methods are simple voting schemes, statistic methods, and fuzzy logic [Rao10, p.7].

The three main levels of fusion can also be seen in Figure 4.2. Since these levels handle different kinds of data, the methods and algorithms that are applied are not the same. Table 4.1 gives a brief overview of the most common techniques [15]. In [Rao10] fuzzy logic, voting, and statistic

Table 4.1: Applied techniques according to the level of fusion [15]

Fusion level	Applied techniques
Raw data	Kalman filtering Inference methods
Feature	Fuzzy logic Neuronal networks Pattern recognition
Decision	Expert systems Artificial intelligence

methods are used at the decision level. It can be concluded that the methods cannot be clearly assigned to one of the fusion levels. Moreover, it is a flexible categorization that is given in Table 4.1. The nature of this behavior is associated with the data origin of each level, which was discussed in Section 4.2.3. So the input and the output data are seen to be the more decisive elements for choosing the appropriate technique.

Despite the common definition of the three levels, [NLF07, p.50] mentioned the concept of *multi-level fusion*. The definition also covers the fact that information from different levels can be fused. A more intuitive classification based on the input-output type is given in the next Section. Fusion will be represented in a hierarchical order.

4.2.3 Input-output type

After defining fusion according to levels of fusion (raw data, feature and decision fusion), Dasarathy goes one step further. In [Das97] he gives a definition of fusion based on the information input and output type. Figure 4.2 shows five classifications with the equivalent levels based on the type of fusion (see right area of the figure).

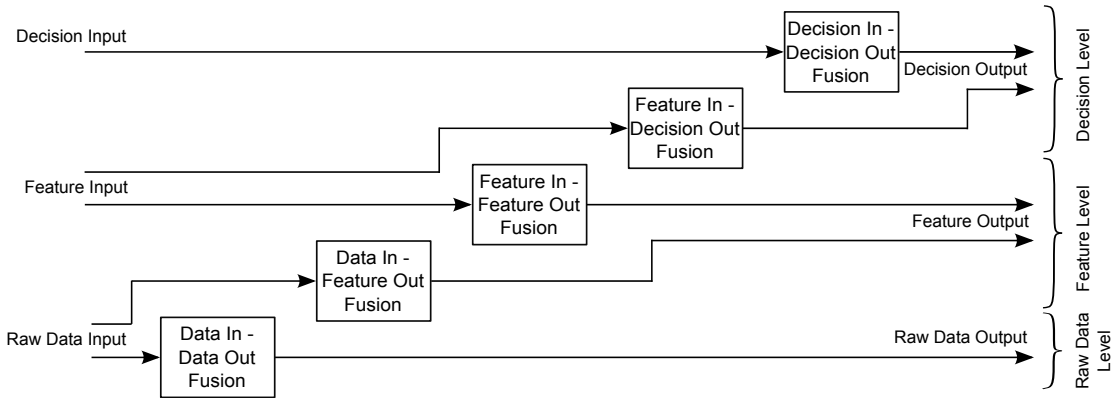


Figure 4.2: Categorization based on the input/output information (adapted from [Elm02])

Data In-Data Out (DAI-DAO) Fusion represents the basic form of fusion. This mode produces a form of output data resulting from input data. The method is commonly referred to as data fusion or raw data fusion. Fusion strategies are often based on techniques used in signal processing or image processing.

Data In-Feature Out (DAI-FEO) Fusion represents the next stage of this five-level hierarchy. Raw data from different sensors are combined to produce a feature, describing an object under observation.

Feature In-Feature Out (FEI-FEO) Fusion usually combines features from various sources to an output feature or to a multidimensional feature space. The processing is used to refine features or to gain new features [NLF07, p.9].

Feature In-Decision Out (FEI-DEO) Fusion can either be arranged in the category of decision fusion or feature fusion. The inputs are features of different sensors. They are finally combined to a decision.

Decision In-Decision Out (DEI-DEO) Fusion is the highest level in the hierarchy (see Figure 4.2) which is also described as the decision fusion process.

Based on the desired output all this fusion methods can be combined. Let's suppose that we start at the raw data level. We have to evaluate the fusion strategies at each level to come to a decision. This can be done by having a kind of serial network of the appropriate fusion modes.

A power supply for example, is proposed including some shut down strategies. In case of too high temperature, over-voltage or over-current the inverter has to shut down. These properties are measured by different sensors. The temperature is measured at different points at heat sinks and at the surrounding area of other critical parts. The voltages and currents are measured at the

input and outputs and other intermediate supply circuits. In case of nearby temperature sensors the data can be combined to a more accurate representation (DAI-DAO fusion). In order to gain a feature, the raw data or data from raw data fusion can be compared to a maximum temperature which is given by the electric components. These features will, for example, represent too high temperature at a heat sink, normal temperature at ambient sensor or over-current at the output converter. These features will be fused (FEI-DEO fusion) to make the decision to power down the inverter.

In reference to the applied fusion strategies of Chapter 5, not all of these levels are progressed by each fusion method. The work is focusing on FEI-DEO fusion and DAI-DAO fusion. DAI-DAO fusion is found when fusing raw data. FEI-DEO fusion applies to the exchange of node classifications and also for deciding a sensor measurement being incorrect.

4.3 Architectures

The choice of the architecture mainly depends on the location at which fusion of the data should take place. The architecture defines how much communication bandwidth, computational complexity, and algorithms are best to use. Furthermore the architecture is responsible for the quality of the fusion product [Sch06]. The three major architectures are *centralized*, *decentralized* and *hierarchical*, which are discussed in the following sections.

4.3.1 Centralized fusion

In a centralized fusion system (see Figure 4.3), a single coordination node is doing the whole fusion process. This involves the collection of the different sensors' data. According to the level of fusion (see Section 4.2.2) raw data, features or decisions are the output of such centralized nodes. According to [Mit07, p.38] a centralized architecture has the best performance if the sensors are

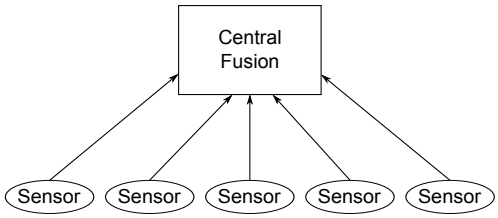


Figure 4.3: Centralized fusion architecture (adapted from [Sch06])

all correctly aligned. However, it also has some disadvantages. In case of high density of sensor nodes the communication to the single fusion centre will cause a communication bottleneck. The architecture is also inflexible [Mit07, p.39] when the application has to be changed. The central processor could lack the computational power to maintain its operation. This could also be possible when the number of sensors is changed. The greatest weakness lies in a central processing node, being the single point of failure either when the communication interface or the processing unit fails.

4.3.2 Decentralized fusion

Figure 4.4 depicts the scheme of a decentralized architecture. The information is fused locally by each node without a central processing unit. Such distributed fusion is mostly used for dissimilar sensor networks but can also be used for similar sensors [Rao10, p.21]. A local node is only fusing its own data with the data of all other nodes or with some surrounding nodes as described in [ZW10]. The main advantages are the easy extensibility of the number of sensors (scalable)

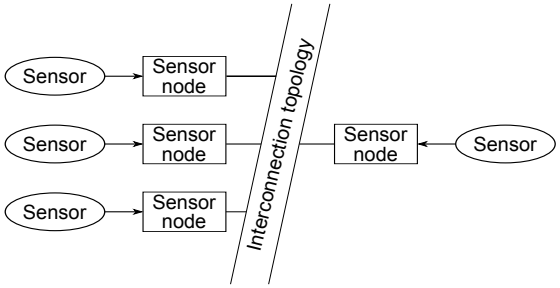


Figure 4.4: Decentralized fusion architecture (adapted from [Sch06])

without changing the underling algorithms or hardware. The robustness in terms of failures of single nodes is one of the benefits. If single nodes fail in their operation, the whole system is still able to continue its operation.

4.3.3 Hierarchical fusion

Hierarchical fusion is a kind of hybrid architecture, combining centralized and decentralized fusion. As depicted in Figure 4.5, data is collected from a small portion of the overall sensor environment and processed in local nodes. Each of the local fusion nodes produces its own output and forwards it to the central processing node. The required computational power and communication bandwidth is thus reduced [Sch06, p.21]. The central processing node finally fuses the results of all subsystems. Even though every sub-node provides its local optimum, it is not guaranteed that

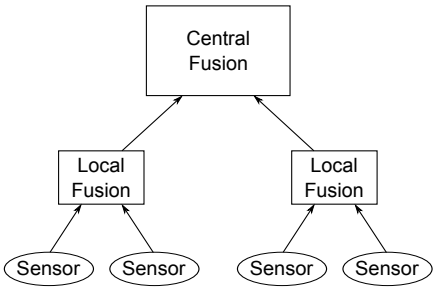


Figure 4.5: Hierarchical fusion architecture (adapted from [Sch06])

the global outcome is the global optimum. The hierarchical structure produces an information loss at each level and doesn't provide a global view.

4.4 Fusion models

In order to have a common methodology to design information fusion systems, several models have been introduced. They have to be kept general in order to cover a lot of application domains and applied methods. The proposed models split the tasks necessary for fusion into smaller pieces. The following sections introduce the reader to some models and their common features.

4.4.1 JDL model

The Joint Directors of Laboratories (JDL) Data Fusion Working Group established this model in 1986. The outcome is a generalized model that fits many applications. It was first developed for military applications but is also applicable to non-military fields [HL97]. This generic architecture

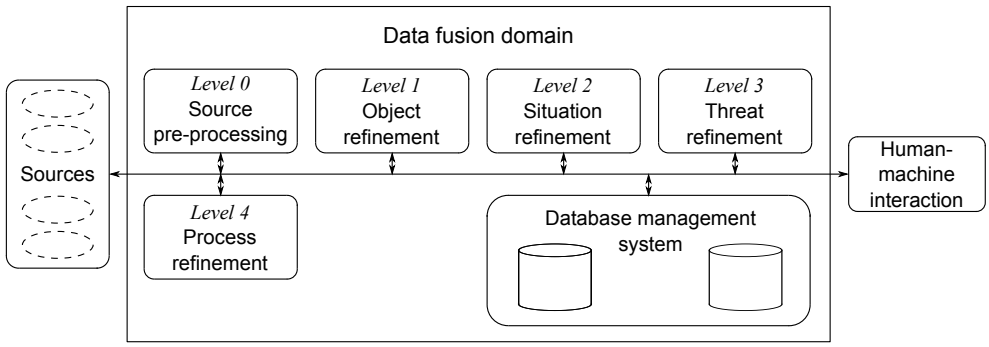


Figure 4.6: JDL model (adapted from [SB01])

is divided into five levels, a database, sources, and a human-machine interaction interface. All components are connected to a bus. Figure 4.6 gives a schematic illustration of the model with all its components. There is no fixed order for the execution of these instances. Moreover, the instances can run concurrently or interleaved. The elements can be described in the following way:

Sources: They provide information from either sensors, prior knowledge from human input or a database.

Source pre-processing (Level 0): It enables the fusion process to concentrate only on the relevant data. Therefore, the data load is reduced. Load reduction is also achieved by assigning reasonable data to the appropriate fusion process.

Object refinement (Level 1): At this level the combination of location, parametric and identity information is done in order to improve the representation of an object. The used techniques are *data alignment*, *association*, *tracking*, and *identification*.

Situation refinement (Level 2): Objects and events become related to each other. In addition, environmental and prior information are taken into consideration to find the relationships.

Threat refinement (Level 3): Level 3 aims to classify the current situation in order to predict possible trends, dangers, and changes for further actions. It's not always easy to predict each outcome, because there are always actions that are not part of the system's sphere of influence.

Process refinement (Level 4): The refinement is a meta-process in which the system performance is observed (e.g. long-term data, real-time constraints). This allows the identification of information which is needed for achieving a particular goal. This way, sensors and other sources can be allocated to the process.

Database management system: The Database management systems support the fusion process by adding, updating, monitoring, evaluating, and offering data.

Human-machine interaction: The human-machine interaction interface offers an interface for the interactions between a person and the machine. Users are able to make inputs such as commands, information requests, assessments of inference, and reports from humans. Moreover, the machine communicates the fusion results to the user or operator. The notification can be done via alarms, displays, or via dynamic overlays of identity and position data on geographic displays.

The JDL data fusion model is a commonly used model but it faces several drawbacks. The model is very useful for many applications but it is hard to identify which actions have to be taken at each level. Therefore they can be misinterpreted very easily. Because of its data-oriented nature, the model does give the interactions but it does not describe which algorithms or methods should be used. Several authors proposed extensions and improvements to these models. In 1998, a revised model that was mentioned in [SBW99] and [SB01] to refine the definitions of each level was introduced. The primary goal is to offer a categorization. This way, different problems can be logically separated. The second objective is to have a consistent terminology. Polychronopoulos and Amditis adapted the JDL model in order to satisfy the needs in the multi-sensor automotive safety systems. In [PA06] they introduce the ProFusion2 (PF2) model. In this functional model they propose a hybrid hierarchical structure for the automotive industry. The document also includes guidelines and recommendations for further implementations.

The relevant levels for this thesis are level 0 to 2, level 4, the database management system and the sources. The information sources are the data sets containing the sensor measurements. The human-machine interface is not considered because the algorithms of Chapter 5 show no interaction with a user.

4.4.2 Waterfall fusion process model

The Waterfall model focuses on the lower processing levels. A functional overview is given in Figure 4.7. The model also shows some similarities to the JDL data fusion model. Sensing and data processing can be matched to level 0 of the JDL model. Feature extraction and pattern processing are equal to level 1, situation assessment fits level 2, and decision making is evident in level 3 [BO00].

The UK defence data community has widely used the Waterfall model but it has never been used considerably elsewhere. A more precise refinement of the individual processing levels is made in this model compared to others. However, it has drawbacks similar to the JDL model. The most

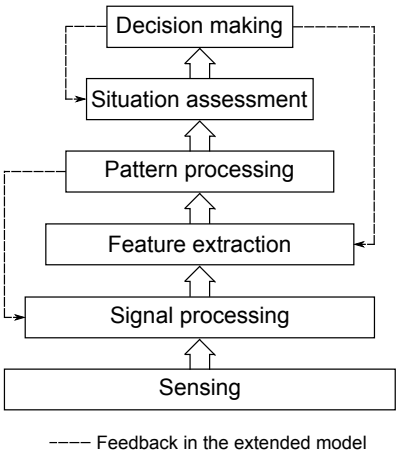


Figure 4.7: Waterfall model (adapted from [BO00])

obvious drawback is that there is no specific feedback loop depicted. In [Rao10, p.17f] a modified version of the Waterfall model is shown. Therefore, some feedback loops have been added. These modifications are also depicted in Figure 4.7. The first doted feedback loop is directed from the decision making process to the situation assessment process which allows to base new decisions on a changed situation. The second control loop from pattern processing to signal processing makes it possible to improve the pattern processing feature extraction and thus the pattern processing. The last feedback is to reflect the refined feature extraction from the current decision. These illustrations indicate that this model is a more action-oriented model compared to the original Waterfall model or the JDL model.

The first step of sensing has not to be evaluated because the provided data set are already containing the sensor data. When fusion raw data, signal processing has to be done only. The node classification algorithm and the combination of the methods have to proceed all further steps in order to make a decision (e.g. correct classification or correct position).

4.4.3 The Boyd Control Loop

The model was formerly used to design military command processes but has later been applied in data fusion processes in a broader context. The Boyd model also refers to the *Observe-Orientate-Decide-Act* (OODA) loop as shown in Figure 4.8. The four phases of the OODA loop do have

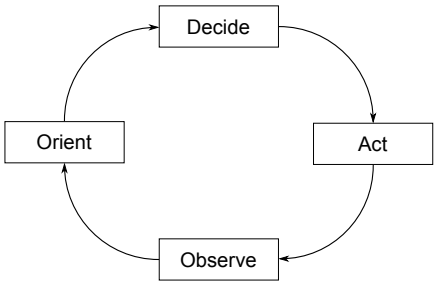


Figure 4.8: Boyd Control Loop (adapted from [BO00])

some features in common with the JDL model as described in the following [BO00]:

Observe: The phase is comparable to level 0 of the JDL model.

Orientate: Level 1, 2 and 3 are the equivalent levels of the JDL model.

Decide: Here, level 4 (process refinement and resource management) can be matched.

Act: A correspondence to the JDL model cannot be found. The phase of acting is just closing the loop which comprises the consequences of decisions.

The relevant phases for the implementations of this thesis are *Observe*, *Orient*, and *Decide*. The phase of acting (*Act*) in AAL systems considers to e.g. raising an alarm which is not part of this work. The model of the next Section is more refined but it shows the same phases like this model. Therefore, the same phases apply to the used fusion algorithms (see Chapter 5).

4.4.4 Omnibus model

The Omnibus model is a combination of different models to overcome their individual shortcomings [BO00]. As can be seen in Figure 4.9, the model looks similar to the Boyd Control Loop. However, like the Waterfall model, this model is more refined. The path between observation and

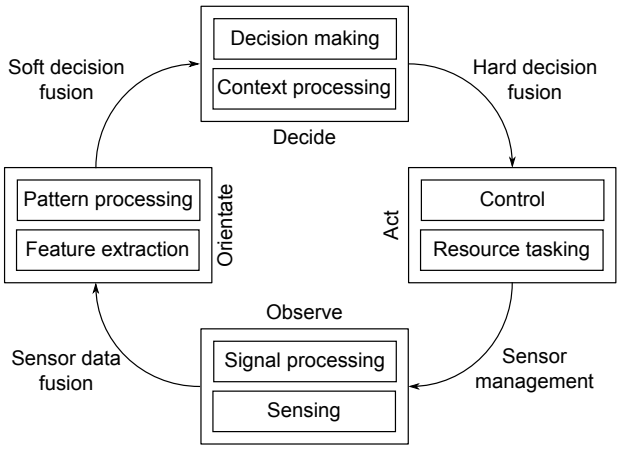


Figure 4.9: Omnibus model (adapted from [BO00])

orientation is referred to as sensor data fusion and the intersection from orientation to decision is considered to be the soft decision fusion. Hard decision fusion is the path from the decision process to acting. The sensor management is done at the path from acting to observation.

The model aims at accomplishing two objectives. The first is to characterize the overall system and divide their tasks into subtasks. Second, the model is used to organize the subtasks. Therefore, the model can be used in order to describe the system and its tasks.

The Omnibus model is more complete than the Boyd Control Loop or the Waterfall fusion model because it combines several aspects of these models.

5 Applied Sensor Fusion algorithms

The previous Chapter introduced its reader to concepts of Sensor Fusion. Different methods can be applied at each fusion level. A precise mapping to one level is not possible. Moreover, the methods will be defined by the intended purpose. Since there are different types of sensors, the measurements have to be treated differently. Thus, algorithms covering continuous measurements (e.g. temperature) will be different from the one with binary outputs (e.g. movement sensors).

For this purpose concepts introduced in Chapter 3 will be adapted to fit the circumstances of the actual application. Due to the assumptions made in the introduction, not all of them can be applied to this work. The Chapter offers a summary of the applied Sensor Fusion algorithms. The theoretical background provides the necessary information for the implementation.

The domain of raw data fusion is covered in Section 5.1. The confidence weighted average fusion algorithm is an algorithm for fusing data from replicate sensors. The demand for fault tolerance is evident in the basic version of this algorithm. Thus, three extensions are also provided.

The second major part of the thesis is devoted to the usability aspect of Ambient Assisted Living systems. It is divided into smaller pieces, covering different aspects. The first method entails the concept of classifying the included sensor types of a node. This means that no further configuration work for the nodes is required in the installation phase. The next algorithm uses data from movement sensors to generate a global representation of a flat. Obtaining the global view is possible without knowledge about the real geometric configurations. Assisted by the person model (see Section 5.4), the global representation is used to determine incorrect sensor activation.

5.1 Confidence weighted average fusion and its extensions

The presented algorithms are based on the confidence weighted average fusion algorithm. The main difference between the implementations is the way how “incorrect” measurements are treated. In order to deal with sensor faults, the following definitions have to be made:

Definition 1: Correct sensor - A sensor is said to work correctly when its value is within an estimated interval. The estimation is based on the observation of other sensors.

Definition 2: Incorrect sensor - An incorrect sensor measurement is supposed present if it is not fulfilling the definition of a correct sensor - the observations of other sensors are not confirming the observations of the faulty sensor.

These definitions are used in the following sections to describe sensor measurements which are evaluated by the algorithms. In this context the terms *erroneous* and *faulty* will also be used.

In [LX10], Lu and Xue present the confidence weighted fusion algorithm. The weighted average of the sensor measurements is calculated according to the uncertainty of each sensor. The precondition to this algorithm is that the sensors are arranged in a replicated way so that each sensor faces the same property. In Section 4.2.1 such a configuration is referred to as *competitive* sensor configuration. The main idea of the algorithm is to assign weights according to the sensors' uncertainties. The uncertainty is expressed by the variance. Sensors having lower variances than others are weighted more and will thus make greater contribution to the fused result x_{Fused}. The sum of the weights w_i has to fulfil the following condition:

$$\sum_{i=1}^{n} w_i = 1 \tag{5.1}$$

when $i = 1, 2 \ldots n$ denotes the sensor number. The variance σ_i^2 is given for each sensor. Therefore, the weights can be calculated as follows

$$w_i = \frac{\frac{1}{\sigma_i^2}}{\sum_{j=1}^{n} \frac{1}{\sigma_j^2}}. \tag{5.2}$$

The expression $\frac{1}{\sigma_i^2}$ in the numerator indicates that the above mentioned assumption - sensors with low variances having higher weights - is true. The fused result is calculated according to

$$x_{Fused} = \sum_{i=1}^{n} w_i x_i \tag{5.3}$$

in which x_i denotes the measured value of sensor i. The fused output value shows a decreased variance σ_{Fused}^2 which calculates as

$$\sigma_{Fused}^2 = \sum_{i=1}^{n} w_i \sigma_i^2 = \frac{1}{\sum_{i=1}^{n} \frac{1}{\sigma_i^2}} \tag{5.4}$$

while the last term is derived from substitution of w_i with the expression of Equation 5.2. When having more sensors $n \geq 2$, the variance of the fused measurement is thus less than the variances of the individual sensors.

In presence of faulty sensors the algorithm has to be handled with care. The weight of an erroneous sensor determines the final result. When the assigned weight of the faulty sensor is low, the influence on the fusion result is also low and decreases with the number of sensors that are used. Considerable weighted incorrect sensor measurements have a particular influence on the fused result. There are several ways to solve this problem. The faulty sensor could be detected by a distance measurement: if the measurement lies within a certain range compared to the other sensors, this sensor can be trusted. Otherwise the sensor is considered faulty [DLC05]. When incorrect measurements are identified, they can be excluded from further application. Elmenreich and Peti propose a fault tolerant extension of the confidence weighted average algorithm in [EP03]. They suggest to neglect the t smallest and t largest values of all measurements. Therefore, t faulty values can be tolerated with at least $2t + 1$ measurements.

5.1.1 Congeneric multi-sensor data fusion

The congeneric multi-sensor data fusion algorithm estimates the sensors' variances based on the fused output. After presenting the iterative equation, the additional memory attenuation factor is introduced. This factor is used to reduce the effect of old data [SQSX10].

The estimated variance is calculated as

$$\hat{\sigma}_s^{*2}(k) = \frac{1}{n-1}\sum_{i=1}^{n}(x_s(i) - x_{Fused}(i-1))^2. \tag{5.5}$$

At k sampling times, n sensor measurements $x_s(i)$ are available there. The fused output $x_{Fused}(k-1)$ at the previous sampling time is known. Equation 5.5 treated the sensors separately and ignored correlations. The next equation gives the iterative expression as

$$\hat{\sigma}_s^{*2}(k) = \frac{k-2}{k-1}\hat{\sigma}_s^{*2}(k-1) + \frac{1}{k-1}\left(x_s(k) - x_{Fused}(k-1)\right)^2. \tag{5.6}$$

The first term considers the estimated variance of the last sampling point. Both pre-factors are considered if the number of samples k tends to infinity. The boundary value of the second term shows that new data will have less effect on the new variances. This way, new data, including some kind of disturbance, can never reflect the actual sensor observations. Therefore, the attenuation factor α is introduced, replacing the factors of Equation 5.6 to the following expression

$$\hat{\sigma}_s^{*2}(k) = \alpha\hat{\sigma}_s^{*2}(k-1) + (1-\alpha)(x_s(k) - x_{Fused}(k-1))^2. \tag{5.7}$$

The memory attenuation factor is in the range of 0 to 1. Using the extreme values, the equation above indicates that only the recent measurements or only the old data is considered. When $\alpha = 1$ is applied, the variance stays the same. Thus, it can be concluded that no difference to the basic confidence weighted average fusion is present.

Prior to the implementation no distinct variances have to be known. The variances will reflect the influence of the individual sensors on the fused output. Therefore, the exact value is not necessary. Moreover, the ratio of the variances, leading to the weights, is important.

5.1.2 Voting algorithm

The approach of [DLC05] has already been introduced in Chapter 3. The concept uses a random elected sensor. The sensor is working correct when a sufficient number of other measurements confirm this sensor. The algorithm terminates after a correct sensor has been found. Thus, using this single sensor means to deliver one value only.

The concept of the distance parameter is reused for this implementation. Compared to the random voting algorithm this algorithm iterates all available sensors and excludes the incorrect ones. Figure 5.1 illustrates that two other measurements confirm the measurement x of sensor 4. The confirming values are in the range of $\pm d$ from x_4. Sensor number 2 did not get any votes and is therefore excluded. The distance parameter d is depending on the sensor's uncertainty and also on the sensor's resolution. The choice of the appropriate distance parameter is given in Chapter 7.1.2.

The confidence weighted average fusion algorithm is used by the sensors collecting enough votes. Using a set of sensors for this algorithm offers the advantage of an increased confidence according to Equation 5.4.

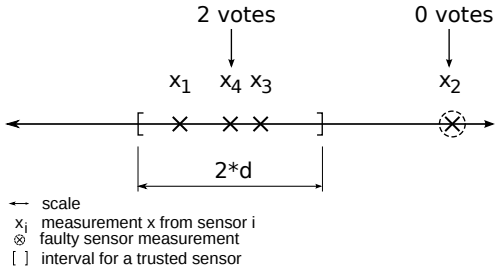


Figure 5.1: Concept of the voting algorithm (d ... distance parameter)

5.1.3 Adaptive algorithm

Inspired by the voting algorithm, the adaptive algorithm does the fusion first and excludes incorrect measurements in the next step. The corrupted measurements are also detected by a distance parameter.

In case of the erroneous measurement x_2, the first scale of Figure 5.2 shows that the fused value x_{Fused} of all measurements is closer to the correct measurements than to x_2. The confirming

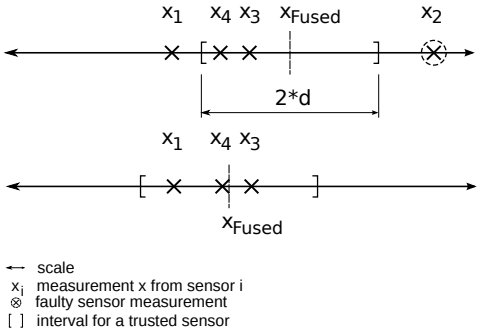


Figure 5.2: Concept of the adaptive algorithm (d ... distance parameter)

measurements draw the fused result towards themselves. Thus, the voting process of the previous algorithm is indirectly given. Consequently, measurement x_2 has the highest deviation from the fused result and can thus be excluded. The fused output is recalculated in the next iteration (see second scale). Now all values are within the interval of $\pm d$, thus terminating the fusion process.

The adaptive algorithm behaves similarly to the voting algorithm, which will be shown in the results of Chapter 7. The influence of different variances and thus weights will also be discussed.

5.2 Node classification

The node classification process addresses the identification of each sensor type, included in a node. The easiest way to do the categorization is to use the sensor values and the data intervals. When this approach is not helpful, additional information is needed. In [Bis95] Bishop illustrates that is very important to use as much information as possible for a precise classification. Therefore, it is practical to study the temporal behavior. Some of the classification criteria are listed in the following:

- Data range
- Event- or time-triggered measurements
- Change of the measurements (abrupt or monotonic)
- Occurrence (frequency, sampling time, temporal distribution)

The analysis of the data shows that the sensor types can be completely described in terms of their data ranges. Therefore, the used sensor measurements can be divided into sensors with continuous scale and those with binary scale (see Figure 5.3). It is quite easy to differentiate among the binary

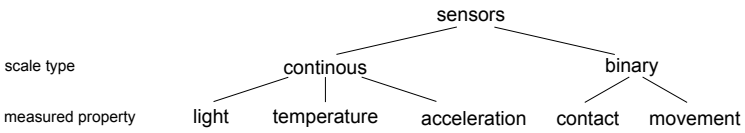


Figure 5.3: Sensor classification

scale types: a reed contact delivers two symbols ("0" and "1") and the movement sensors just offer the symbol "1" when there is any movement.

The sensor types with continuous scale can be determined by their sensor intervals. It is a more difficult task because of the overlapping intervals. Figure 5.4 displays these overlapping intervals. The intervals I_i depict the observed data intervals and the intervals I_i' give the data intervals

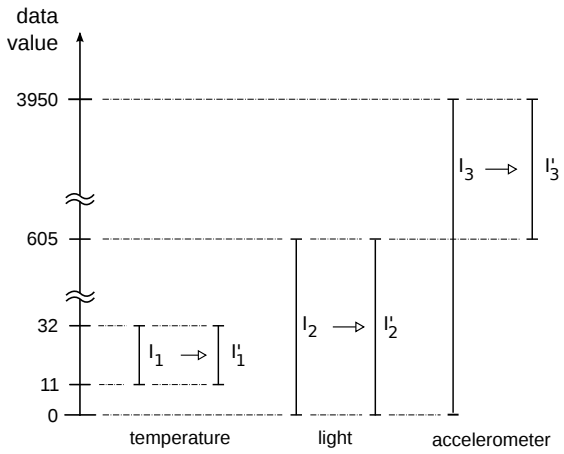


Figure 5.4: Overlapping intervals of different sensor types (I_i ... data interval of sensor i, I_i' ... data interval for the classification process)

for further implementation. With reference to the used sensor types, interval I_1' displays the data range of the temperature sensor, I_2' the interval of the light sensor and the third interval represents the data range of the accelerometers. The indices also indicate the scanning order at the classification process.

The interval I_1' lies within I_2'. Therefore, I_1' has to be checked first. The none-overlapping interval I_3' is evaluated after the overlapping intervals of I_1' and I_2' are checked.

At this point the benefits of Sensor Fusion are not faced at all. The gained sensor classifications have to be exchanged in order to find a global consensus. A specific sensor-ID presenting the same type in each node is assumed. It is also preconditioned that the sensor type depicted by the sensor-ID is not known before. Figure 5.5 shows that n nodes submit their individual classifications to sensor-ID 3. Node 3 identified an incorrect classification. From the consideration of the other

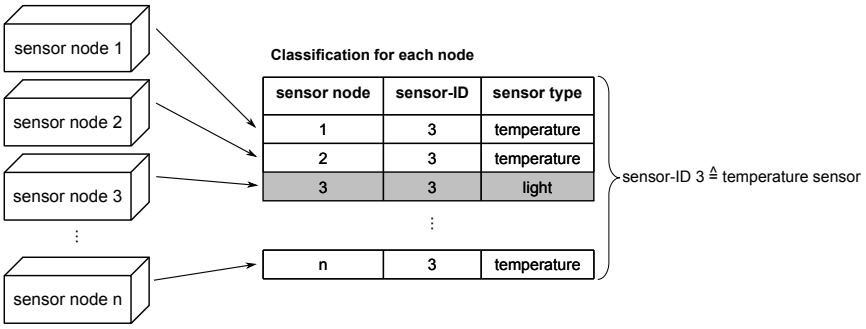


Figure 5.5: Sensor type classification by consensus

nodes, however, it is concluded that sensor-ID 3 represents a temperature sensor. All incorrect classifications are replaced in a final step.

5.3 Topology learning

The basic idea for topology learning with the help of data from movement sensors has already been presented in Chapter 3. The compatibility matrices are used to demonstrate the correlations between the states of two nodes. For this application the vector, representing the states of [PCH08], reduces its dimension to a scalar. The single state indicates an observed movement.

The 2×2 compatibility matrix covers all state combinations of a sensor pair $(i,\ j)$. Using the two symbols S_i and S_j, the compatibility matrix is written as

$$\Psi_{ij}(S_i, S_j) = \begin{bmatrix} \Psi_{ij}(S_i = 0, S_j = 0) & \Psi_{ij}(S_i = 0, S_j = 1) \\ \Psi_{ij}(S_i = 1, S_j = 0) & \Psi_{ij}(S_i = 1, S_j = 1) \end{bmatrix}. \tag{5.8}$$

The main diagonal indicated that both nodes observe a movement or no movement at the same time. The secondary diagonal expresses the situation when only one of the nodes observes a movement. The compatibility matrix is normalized to limit the data values. Thus, the entries represent the probabilities for each event.

The correlations are determined in the next step. The authors of [PCH08] suggest that a correlated sensor pair is given when the main diagonal is dominant (see Equation 5.9). Correlations are not faced in case of a dominant secondary diagonal or if all entries of the matrix have approximately the same value (see Equation 5.10).

$$\Psi_{ij,corr} = \begin{bmatrix} 0.1 & 0.01 \\ 0.09 & 0.8 \end{bmatrix} \tag{5.9}$$

$$\Psi_{ij,uncorr} = \begin{bmatrix} 0.17 & 0.50 \\ 0.01 & 0.22 \end{bmatrix} \tag{5.10}$$

In the paper it was not explicitly explained when correlation is to be considered assured because these matrix entries are in an additional operational step used as weights [PCH08].

The addressed issue finds its implementation in the comparing of the sum of the main diagonal to the sum of the secondary diagonal. The circumstance of correlation is achieved when a certain ratio between these expressions is being exceeded. Further implementation details can be found in Section 6.2.2.

5.4 Person model

The person model is inspired by the work of [Mit08]. In this work it is a tool for further processing steps, which allows to eliminate false positive sensor activations.

The purpose of the person model is to determine the actual position of a person. The current location is governed by the movement sensors. These spatial data, with regard to the aspect of time, reflect the behaviour of the person. Usually, there is just one citizen in an Ambient Assisted Living environment. Thus, it is sufficient to determine the position of one person only.

Figure 5.6 shows a typical case when a person is recognized by two sensors. The two sensors (S1 and S2) observe the movement of the person P at approximately the same time. The active

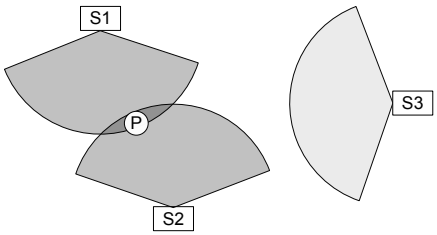


Figure 5.6: Capturing a person's position with movement sensors (the segments of the circles indicate the sensor coverage; gray... active sensor, light gray... inactive sensor, P... person, S... sensor)

sensors are indicated by the gray coverage area while an inactive sensor S3 is indicated by the light gray area. In the representation of the person, the two sensors (S1 and S2) are assigned

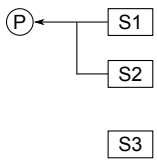


Figure 5.7: Person model including the assigned sensors

to the person model (see Figure 5.7). The current sensors can be released when new data is available. A sensor that is already present is maintained in the sensor list of the person model.

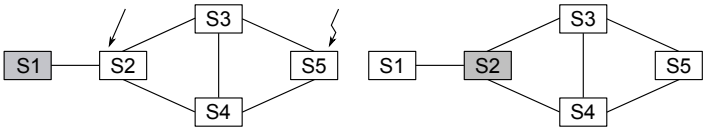
The tool offers the opportunity of determining the person's location based on the sensor activations. False positive observations influence the whole process. Thus, the next section introduces the reader to fault tolerance.

5.5 Combining of the methods

The combination of the node classification process with topology and the person model is demonstrated. The node classification delivers all movement sensors which are used to discover the topology of a flat. The core function is gained from the combination of the person model and the topology. The person model comprises the actual position. The topology indicates if new positions can be reached from the actual one.

Therefore, some possible scenarios of sensor faults are given. The movement sensors are depicted in the rectangular boxes, including the sensor numbers Sx. The connections between the nodes show which sensors (areas) can be reached from the current sensor. In the person model the assigned sensors are indicated by the gray shading of the sensors.

A typical topology, including one correct activation and one incorrect activation, is depicted in Figure 5.8. The second sub-figure shows the effect of these activations. The person model holds

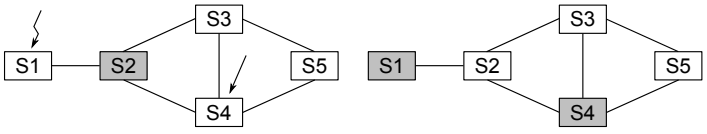


(a) Sensor activations (b) Resulting person model

Figure 5.8: Topology with correctable sensor fault: actual sensor activation (a) and the resulting activations (b) (gray... sensor assigned to person model, S... sensor, arrow... correct activation, flash... incorrect activation)

the sensor S1. Two activations are observed. Sensor S5 cannot be reached from sensor S1. Thus, it is neglected. The topology shows that S2 can be reached from sensor S1. Figure 5.8(b) displays the resulting sensor that is included in the person model. False positive events can be detected as long as at least one node is between the current sensor node and the faulty one.

The worst case situation is illustrated in Figure 5.9. The faulty and the correct node can be reached from the correct one. So a differentiation between those nodes is not possible. The



(a) Sensor activations (b) Resulting person model

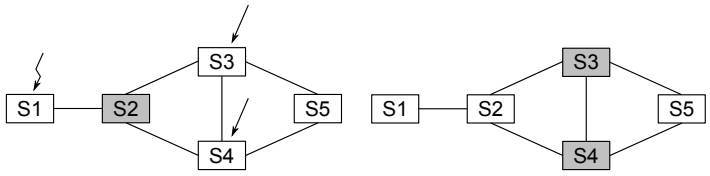
Figure 5.9: Topology with non-correctable sensor fault: actual sensor activation (a) and the resulting activations (b) (gray... sensor assigned to person model, S... sensor, arrow... correct activation, flash... incorrect activation)

resulting person model will comprise both sensors (S1 and S4) (see Figure 5.9(b)). Additional correct sensor activations help to re-establish an accurate person model.

A person model that gets trapped in an area is another scenario. The situation is caused by incorrect sensor readings. Sensor S1 of Figure 5.9(a) shows another incorrect activation. Therefore, the next person model would only include that sensor. When having correct sensors that are not reachable, these sensors cannot be added to the person model. The issue of getting trapped is

solved by an additional counter. The count indicates the number of "wrong" activations. The reset of the person model happens when the counter reaches a specified limit. The same problem is solved when the person gets to a reachable node again before the counter has reached its reset count.

Another scenario is shown in Figure 5.10. Sensor S3 delivers the information of movement. Sensor

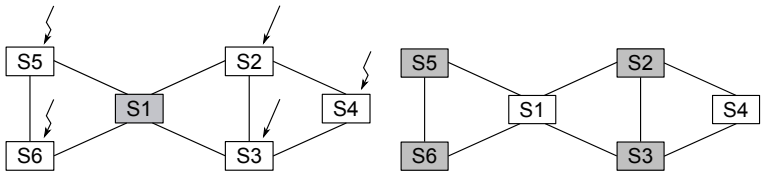


(a) Sensor activations (b) Resulting person model

Figure 5.10: Topology with probably correctable sensor fault: actual sensor activation (a) and the resulting activations (b) (gray. . . sensor assigned to person model, S. . . sensor, arrow. . . correct activation, flash. . . incorrect activation)

S3 and S4 are linked together and can be reached from the current sensor node. Contrary to these sensors, sensor S1 is not linked to these sensors but can be reached from S2. It is more likely that the two activations from the linked sensors are correct than the event from the single sensor. Therefore, the single sensor is not added to the person model (see Figure 5.10(b)).

The last scenario that will be presented, is depicted in Figure 5.11. It is a combination of the situations depicted in Figure 5.9 and 5.10. The person model includes the current active node S1. All other sensors are facing events, regardless of whether they are correct or not. The reachable



(a) Sensor activations (b) Resulting person model

Figure 5.11: Topology with transient uncorrectable sensor faults: actual sensor activation (a) and the resulting activations (b) (gray. . . sensor assigned to person model, S. . . sensor, arrow. . . correct activation, flash. . . incorrect activation)

nodes represent two linked groups. These groups have the same size. Thus, they are considered in the next iteration step. The person model includes a reachable sensor node (see Figure 5.11(b)). If the two groups differ in size, the larger group is taken into account for the person model (see Figure 5.10). A detailed description of how to implement the mentioned scenarios can be found in Section 6.2.4.

6 Implementation

The previous Chapter has introduced the reader to the theoretical background of the applicable fusion algorithms. In this Section these algorithms are implemented. The programming environment MATLAB is used for the implementation. Specific constructs are presented that make the implementation easier. Additional graphics and explanations will also be given to demonstrate some implementation details.

Two separate fields have already been identified - one covers the opportunity to fuse raw data, the other covers the minimization of configuration work paired with fault tolerance.

6.1 Raw data fusion

In the theoretical part of Chapter 5 the algorithms of weighted fusion have been mentioned. The pseudo-code for each implementation is provided.

The weighted fusion algorithm and its extensions are used to fuse continuous data. Four different versions of this algorithm are implemented. First of all, the basic confidence weighted fusion algorithm, as it is presented in Section 5.1, is used. It does not contain fault tolerant behaviour. Moreover, the three other algorithms fulfil this condition. The first is a voting algorithm: it classifies a sensor as either faulty or correct according to the complementary sensor measurements of other sensors. A correct sensor must receive the majority of all voters. Another algorithm performs the fusion process at first. According to the fused result, erroneous sensors are excluded from the algorithm and perform the fusion until no more faulty sensors are participating in the process. The last algorithm performs a weighted fusion algorithm in which the variances are re-calculated every iteration.

6.1.1 Confidence weighted average fusion

The confidence weighted average fusion algorithm can be used for fusing raw data as shown in Figure 4.2. The following pseudo-codes will present the implementations in MATLAB. The confidence weighted average fusion algorithm is given in Algorithm 6.1. The variance (*variances*) and measurement (*dataValues*) are the input parameters. The measurements and variances are given in arrays in which the respective sensors are addressed by the same index in each array (see Figure 6.1).

Using MATLAB, array calculations can be done in just one step. This way, the inverse of the variances are evaluated in the first program line. The dot before the operator ("/") indicates that the operation is executed element-wise. In line 2 the sum of the inverse variances are calculated which is expressed in the denominator of Equation 5.2. The two slashes ("//") are the indication of a comment. According to Equation 5.3 the fused output is obtained after calculating the weights. The algorithm has to be performed whenever new data values are available.

Algorithm 6.1: confidenceWeightedAvgFusion()

Input: $dataValues$ - measurement values, $variances$ - sensor variances
Output: x_{Fused} - fused result

1: $invVar = 1./variances$ // inverse variances
2: $sumInVar = sum(invVar)$
3: $weights = invVar/sumInVar$
4: $x_{Fused} = sum(dataValues. * weights)$

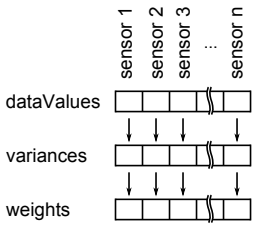


Figure 6.1: Corresponding array indices of *dataValues*, *variances*, and *weights* (n. . . number of sensors)

6.1.2 Congeneric multi-sensor data fusion

The congeneric multi-sensor data fusion algorithm has been discussed in Section 5.1.1. The variances and the weights are re-calculated for each sampling point. Algorithm 6.2 gives the implementation steps. The initialization starts with the variances *initialVariances* and the

Algorithm 6.2: congenericFusion()

Input: $dataValuesDay$ - sensor measurements of a single day, $initialVariances$ - initial values of variances, $x_{Fused,init}$ - initial fusion value, α - attenuation factor
Output: x_{Fused} - fused result

$variances = initialVariances$
$x_{Fused} = x_{Fused,init}$ // set all initial values
for all $dataValuesDay$ **do**
  get current measurements **and** store them in $dataValues$
  $variances = \alpha. * variances + (1 - \alpha) * (dataValues - x_{Fused})^2$ // new variances
  $x_{Fused} = confidenceWeigthedAvgFusion(dataValues, variances)$ // for the current measurement
end for

initial value of the fused output $x_{Fused,init}$. The fused output value (x_{Fused}) is calculated in every

instance of time (line 3). According to Equation 5.7 all variances are calculated in a single step (line 5). The next code evaluates the confidence weighted average algorithm which is presented in Algorithm 6.1.

6.1.3 Voting algorithm

The implementation of the voting algorithm is presented which is not depending on past results. Thus, just one step to produce the fused output is depicted. Algorithm 6.3 shows the pseudo-code to find correct sensor measurements. All sensor measurements x_i are given at an instance of

Algorithm 6.3: votingAlgorithm()

Input: x_i - all measurements, i = 1,2..., n - number of sensors, d - distance parameter, v - number of votes for classifying a measurement as correct

Output: $correctSensorsMeasurement$ - correct measurements

1: **for** $i = 1 \rightarrow n$ **do**
2: $x = x_i$
3: x_{Others} = all measurments excluding x_i
4: $votes = sum(|x - x_{Others}| \leq d)$
5: **if** $votes \geq v$ **then**
6: classify x_i as $correctSensorsMeasurement$
7: **end if**
8: **end for**

time: i being the sensor number ($i = 1, 2 \ldots, n$) and n being the total number of sensors. The distance parameter (d) and the number of votes (v) that classify a sensor as correct are also input parameters. The expression x_{Others} of line 3 is an array, excluding the current measurement x_i from all measurements. Using MATLAB, the expression ($|x - x_{Others}| \leq d$) (line 4) gives the indices (logical expressions) of the measurements confirming x_i. The sum of these logical expressions gives the number of votes. Therefore, a sensor measurement can be classified as correct if the *votes* are equal or exceed the value of the predefined bound v. After all correct sensor measurements are known, the confidence weighted average fusion algorithm is performed by using Algorithm 6.1.

6.1.4 Adaptive algorithm

The adaptive algorithm, as presented in Algorithm 6.4, fuses all measurements at first. In an iterative way (line 1 to 6), all measurements x_i are compared to the fused value x_{Fused}. All measurements that deviate more than $\pm d$ from x_{Fused} are determined (line 3). The measurement which deviates the most from the fused value is excluded (see Figure 5.2). In the next iteration the fused value is re-calculated. The procedure is continued until all sensor measurements are within the defined range of $\pm d$ of the fused value x_{Fused}.

6.2 Algorithms increasing usability

The second major part of this thesis is to focus on the usability of the system. Therefore, the next sections discuss the implementation of the node classification process, learning a flat's topology,

Algorithm 6.4: adaptiveAlgorithm()

Input: x_i - all measurements, i = 1,2..., n - number of sensors, d - distance parameter
Output: x_{Fused} - fused output

```
1: repeat
2:    x_Fused = confidenceWeightedAvgFusion()
3:    if sensor(s) have a higher deviation from x_Fused than d then
4:       exclude measurement with the highest deviation
5:    end if
6: until all measurements are within ±d from x_Fused
```

the person model and the combination of the applied methods. All topics have already been introduced in the respective sections of Chapter 5.

6.2.1 Node Classification

The node classification process consists of several steps. All sensor types of each node are classified in the first step. Using the data of a certain interval offers a considerable amount of sensor data. The data format was discussed in Section 2.3, including important information about node number (*nodeID*), sub-type (*SubType*), and data value (*Val0*).

At first, every sensor node has to get to know its included sensor types. In a second step this information is exchanged among the other nodes. The individual classifications will not be correct all the time. The decision-making via consensus will lead to an extended view of all nodes. Thus, better classification rates[1] should be achieved. This statement will have to be investigated in the respective part of the results of Chapter 7.

Algorithm 6.5 starts with the basic steps of writing the node classifications. The data (*sensorData*) of a certain interval is collected and analyzed in order to find all sensor numbers (*nodeNumbers*) (line 2). The time interval determines how many of the nodes and sub-types can be identified.

Algorithm 6.5: Writing the classifications for all nodes

```
1: sensorData = data from interval
2: get all nodeNumbers from this interval
3: for all nodeNumbers do
4:    subtype-IDs= getSubtypeIDs(nodeNumber) // get all subtype-IDs of a node
5:    for all subtype-IDs do
6:       add the result of classifyType() to the list of classifiedTypes
7:    end for
8:    add classification of each type to sensorExchangeData including nodeNumber,
      subtyp-eIDs and classifiedTypes
9: end for
```

Therefore, each data entry includes the node number which is used to identify the sensor nodes and the sub-type number. For every node number all sub-types are determined by using the

[1] classification rate... ratio of correct classifications to all classifications

function *getSubtypeIDs*. These sub-types are classified (*classifiedTypes*) in line 6. After the classification, the *nodeNumber*, *subtype-IDs* and the corresponding *classifiedTypes* are written into the cell array *sensorExchangeData*. This array includes the representation of all nodes.

Algorithm 6.6 demonstrates the classification of one individual sensor type. Therefore, the function requires the sensor data (*sensorData*) of one sensor type of a node. The recording period of the data is usually one day but this interval is part of the discussion in Chapter 7. In line 1 and 2 the data interval is determined by the maximum and minimum value of all records. In Section 5.2 it was highlighted that binary sensors deliver the symbols "1" or "0." Moreover, a movement sensor only indicates the movement by sending the symbol "1." These conditions are covered by program lines 3 to 8. The temperature sensor, light sensor, and accelerometer are classified by their data ranges. As a consequence of Figure 5.4, the depicted order of scanned the intervals (I_i') is evaluated in the code lines 10 to 18. The $typeClassification$ - expressions such as *temperatureSensor* are integer values. The assigned value is not important. Moreover, the number has to be unique.

Algorithm 6.6: classifyType()

Input: *sensorData* - collected sensor values of a node's sub-type
Output: $typeClassification$ - classification for the current sensor type

```
1:  maxValue = max(sensorData)
2:  minValue = min(sensorData)
3:  if maxValue == 1 then
4:    if maxValue == minValue then
5:      typeClassification = pirSensor
6:    else
7:      typeClassification = reedSensor
8:    end if
9:  else
10:   if maxValue and minValue are within temperature sensor limits then
11:     typeClassification = temperatureSensor
12:   else
13:     if maxValue and minValue are within light sensor limits then
14:       typeClassification = lightSensor
15:     else
16:       typeClassification = accelerationSensor
17:     end if
18:   end if
19: end if
```

The next step is to find a consensus on all classifications stored in *sensorExchangeData*. This behavior is realized in Algorithm 6.7. Using the gained classifications of all sensor nodes of *sensorExchangeData*, all contained sub-types are determined and sorted without including replicas (line 1 to 2).

Now the classifications for every subtype (*subtype-ID*) are collected and stored into a cell array of *votingClass* (code line 3 to 15). Figure 6.2 depicts the data structure of the cell array of *votingClass*. This array has the same size (k) as there are number of sub-types. The entries of the array include the sensor-type (*subtype-ID*) and individual classifications of each node.

After pre-processing the intermediate sensor classification of all nodes, the majority voting process is applied to get the correct type classifications. The outputs of this algorithm are two arrays (*typeNumber* and *typeClass*). Where *typeNumber* contains all *subtype-IDs* and *typeClass* contains the classification which is represented by an integer value. These two arrays are of identical size. Using the same array indices, the subtype-ID with its corresponding classification is obtained. It is the same indexing procedure as presented in Figure 6.1. Finally, the classification of the individual

Algorithm 6.7: votingClassification()

Input: sensorExchangeData - arrays containing $nodeNumber$, $subtype\text{-}IDs$ and $classifiedTypes$

Output: typeNumber - equal to $sensor\text{-}ID$, $typeClass$ - classified types by majority voting

```
1: get all subtype-IDs contained in sensorExchangeData
2: subtype-IDs = unique(subtype-IDs) // sort the numbers and exclude replicates
3: for all subtype-IDs do
4:    votingData = [ ] // initialize
5:    for all sensorExchangeData do
6:       find subtype-ID
7:       if nothing found then
8:          continue
9:       else
10:         add the classification from classifedTypes of this subtype-ID to votingData
11:      end if
12:   end for
13:   write newData containing subType-ID and votingData
14:   add newData to votingClass
15: end for
16: for all votingClass do
17:   do majority voting to get the classType for the subType-ID
18: end for
```

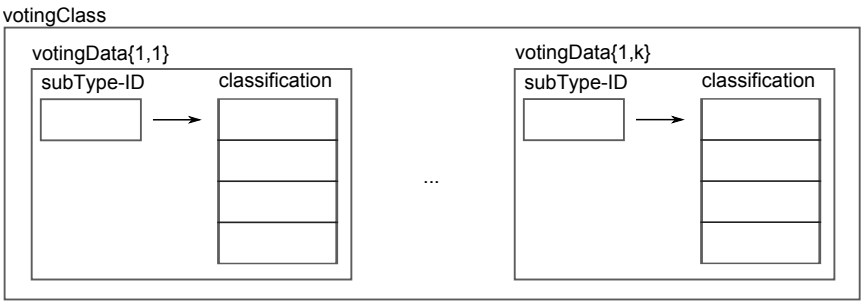


Figure 6.2: Data structure of *votingClass* (k... number of different sub-types)

sensors (*sensorExchangeData*) can be replaced by the more advanced version (*classifiedSensors*). Algorithm 6.8 progresses all nodes and replaces the *classifiedTypes* with the correct types from *typeClass*.

The data of *classifiedSensors* includes the classifications which do not have to be the correct ones. The quality in terms of classification rate will be discussed in Section 7.2.

Algorithm 6.8: writeClassifications()

Input: sensorExchangeData - arrays containing $nodeNumber$, $subtype\text{-}IDs$ and $classifiedTypes$; typeNumber - equal to $sensor - ID$, $typeClass$, $typeClass$ - classified types by majority voting

Output: $classifiedSensors$ - sensor classification by consensus

1: **for all** $sensorExchangeData$ **do**
2: find($sensor\text{-}ID == typeNumber$) **and** replace the $classifedType$ with the correct $typeClass$
3: add the correct classified sensor to $classifiedSensors$
4: **end for**

6.2.2 Topology learning

The topology learning algorithm is based on the theoretical concept from [PCH08]. It is adapted to fit the learning of the topology of a flat equipped with movement sensors.

The following algorithms will give the implementation details. The whole process can be divided into the following steps: pre-processing, calculation of the compatibility matrices, and the classification of the individual matrix entries for correlation.

Algorithm 6.9 shows the necessary steps for pre-processing the data. The data from the movement sensors are used when only one person is present in the flat. In the collected data a list of sensor activations is included. Only the transitions are relevant for the used algorithms. If a person

Algorithm 6.9: sensor value preprocessing for the topology

1: get all movement sensors
2: load the data from these movement sensors when just the test person is present
3: get the transitions // the data when change happens
4: **for all** $nodeNumbers$ **do**
5: insertMovementBloking() // insert zeros
6: **end for**

remains in a certain area in which only one sensor is installed, this single sensor produces a set of sequential events. Transitions are obtained when the values between the first and the last value of a sequence are neglected. The transitions are gained after executing line 3.

The movement sensors only provide the symbol “1” when movement is observed. Thus, zeros have to be inserted in order to provide the appropriate information for the compatibility matrices. As Figure 6.3 shows, a zero is inserted after the blocking time ($blockingTime$) of the sensor. When two consecutive events are in close time relationship, the period for a positive activation is extended. Zeros are only inserted if two events have a larger time difference than the maximum time interval ($maxTimeInterval$). This additional action avoids too many zeros which would produce less correlated events.

The pseudo-code for the illustration of Figure 6.3 is given in Algorithm 6.10. Every $sensorData$ entry contains the time stamp, the data value (“1”), and the node number. Therefore, the time difference between two events called $deltaTime$ is calculated. If the time difference is larger or equal to the maximum time interval ($maxTimeInterval$), a data entry consisting of $nextTimeStamp$, 0, and node number is added to $blockingValues$. After the iterations the array of

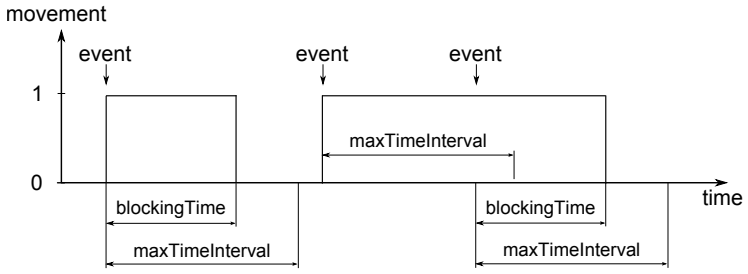


Figure 6.3: Illustration of the function *insertMovementBlocking*

blockingValues, containing the zeros, is gained. The data (*blockingData*) is obtained in line 9 and 10 by merging and sorting *blockingValues* and *sensorData* in accordance with their timestamps. The input of this function is the data of a single sensor node. The procedure is done for each node at all given days.

Algorithm 6.10: insertMovementBlocking()

Input: $sensorData$ - time stamp, data value, and $nodeNumber$; $blockingTime$ - time until no new event can happen; $maxTimeInterval$ - maximum time between two events without inserting a zero

Output: $blockingData$ - $sensorData$ including zeros after events

1: **for all** $sensorData$ except the last one **do**
2: $timestamp$ = get the time stamp of the current $sensorData$
3: $nextTimeStamp = timestamp + blockingTime$
4: $deltaTime$ = the next event and the current $timestamp$
5: **if** $deltaTime \geq maxTimeInterval$ **then**
6: add a zero at $nextTimeStamp$ and the node number to $blockingValues$
7: **end if**
8: **end for**
9: $blockingDataTmp$ = the data from $sensorData$ and $blockingValues$
10: $blockingData$ = sort $blockingDataTmp$ by $timestamps$

The next step is to set up the compatibility matrices for each day. These matrices have to be evaluated for each sensor combination. The gained construct is a matrix, each element being a matrix itself. Equation 6.1 depicts this construct as follows:

$$\Psi = \begin{pmatrix} \Psi_{11} & \cdots & \Psi_{1j} \\ \vdots & \ddots & \vdots \\ \Psi_{i1} & \cdots & \Psi_{ij} \end{pmatrix} \tag{6.1}$$

i and j are the corresponding indices of the sensor nodes. The elements of the compatibility matrix are entered as highlighted in Equation 5.8. Algorithm 6.11 shows the steps for calculating the compatibility matrix of a sensor pair on a single day. Therefore, the gained data (*blockingData*) from Algorithm 6.10 are used. At first, this data is aligned with each other to get a representative format. The first column of the aligned data (*alignedData*) depicts the actual time stamp, the next one the sensor value of the sensor node i, and the third column the sensor value of node j. This algorithm is not presented because it is similar to Algorithm 6.10 with regard to behavior.

Algorithm 6.11: calculating the compatibility matrix of a sensor pair

1: get the *blockingData* for the sensor pair
2: *alignedData* = align this data according to their timestamps
3: $compMatrix = getBoltzCompMatrix()$

Without using the time stamp, the data shows the combinations of [0 0], [0 1], [1 0], and [1 1]. These pairwise events are used in the function $getBoltzCompMatrix()$ (see Algorithm 6.12) to calculate the compatibility matrix. The algorithm starts with an empty matrix containing

Algorithm 6.12: getBoltzCompMatrix()

Input: *alignedData* - time stamp and data value of node i and j
Output: $compMatrix$ - compatibility matrix
1: $compMatrix = zeros(2, 2)$ // initialize an empty matrix
2: **for all** *timeStamps* **do**
3: *currentData* = get the data values of *alignedData* at the current *timeStamp*
4: **if** *currentData* == [0 0] **then**
5: $compMatrix(1, 1) = compMatrix(1, 1) + 1$
6: **else**
7: **if** currentData == [0 1] **then**
8: $compMatrix(1, 2) = compMatrix(1, 2) + 1$
9: **else**
10: **if** currentData == [1 0] **then**
11: $compMatrix(2, 1) = compMatrix(2, 1) + 1$
12: **else**
13: $compMatrix(2, 2) = compMatrix(2, 2) + 1$
14: **end if**
15: **end if**
16: **end if**
17: **end for**

zeros. The data from a single day (*alignedData*) are iterated, finding the mentioned activation combinations. The cell values of the compatibility matrix are increased when the respective events happen. Moreover, the matrix elements contain the number of activations, observed on that day.

When having m days of observations, all gained matrices Ψ_k have to be combined. The corresponding compatibility matrices have to be added to a complete representation as in the following:

$$\Psi_{sum} = \begin{pmatrix} \sum_{k=1}^{m} \Psi_{11,k} & \cdots & \sum_{k=1}^{m} \Psi_{1j,k} \\ \vdots & \ddots & \vdots \\ \sum_{k=1}^{m} \Psi_{i1,k} & \cdots & \sum_{k=1}^{m} \Psi_{ij,k} \end{pmatrix} \tag{6.2}$$

when $k = 1, 2, \ldots, m$ denotes the actual day. The obtained sums of the compatibility matrices are normalized (the entries of the compatibility matrix amount to 1) in the same step, which is useful in the final step of finding the correlations between the sensors.

The implementation in Algorithm 6.13 compares the sum of the main diagonal (trace) to the sum of the secondary diagonal. If the ratio *diagRatio* between *sumMainDiag* and *sumSecDiag*

Algorithm 6.13: classifyCompMatrix()

Input: $compMatrix$ - compatibility matrix, $thresholdRatio$ - ratio to decide if correlated or uncorrelated
Output: $compMatrixClassification$ - classification of $compMatrix$

```
1:  sumMainDiag = trace(compMatrix)
2:  sumSecDiag = trace(fliplr(compMatrix)) // sum of the secondary diagonal
3:  if sumSecDiag == 0 then
4:      compMatrixClassification = 1
5:  else
6:      diagRatio = sumMainDiag/sumSecDiag
7:      if diagRatio ≥ thresholdRatio then
8:          compMatrixClassification = 1
9:      else
10:         compMatrixClassification = 0
11:     end if
12: end if
```

(line 6 to 11) exceeds the predefined threshold ($thresholdRatio$), the sensor pair correlates. The code in lines 3 to 4 covers the case when the entries of the secondary diagonal are zero. This observation is made when the correlation of a sensor with itself is calculated.

The classification process reduces the complexity of the data significantly. Instead of the matrix containing matrices, the matrices got replaced by scalar values. The algorithm 6.13 shows that a correlated sensor pair is encoded by the symbol "1" and the uncorrelated sensor pair by the symbol "0." The obtained matrix is referred to as the topology or the topology matrix in further sections.

6.2.3 Person Model

The representation of a person is achieved by using a structure that includes the identification number of a person and the active sensors. The person number is not explicitly needed but is used to discriminate between the test persons. The following pseudo-codes offer a few simple functions for using this model.

The first Algorithm (see Algorithm 6.14) initializes the person model with the current number of the test person. In the structure of the person model an empty array is initialized which is meant to include further sensor activations. The sensors are assigned by using the function

Algorithm 6.14: initPerson()

Input: $personNumber$ - reference number of the person
Output: $personModel$ - person model

```
1: personModel.number = personNumber
2: personModel.sensors = [ ] // empty array
```

addSensorToPerson. The pseudo-code is given in Algorithm 6.15. The *unique*-command ensures that each sensor number is present only once. The same command also sorts the array of the assigned sensors. The function *overrideSensorsOfPerson* of Algorithm 6.16 completely overrides

Algorithm 6.15: addSensorToPerson()

Input: $personModel$ - current person model, $activeSensors$ - sensor numbers of the active sensors

Output: $updatePersonModel$ - updated person model

1: $updatePersonModel = personModel$
2: $updatePersonModel.sensors = unique([updatePersonModel.sensors\ activeSensors])$

all current sensors of the person model. Furthermore, the introduced algorithms are used in combination with the topology of a flat which is presented in the next Section.

Algorithm 6.16: overrideSensorsOfPerson()

Input: $personModel$ - current person model, $activeSensors$ - sensor numbers of the active sensors

Output: $updatePersonModel$ - updated person model

1: $updatePersonModel = personModel$
2: $updatePersonModel.sensors = unique(activeSensors)$

6.2.4 Combination of the methods

The combination of the methods was presented in detail in Chapter 5. First, the classification is done in order to find all movement sensors. In the next step the topology of the room is learnt. As already described in Section 6.2.2, a matrix results from this process. This matrix determines the linked sensors.

The following implementations describe the combining of the person model with the topology. The gained topology is loaded in an initial step. The person model is initialized with the current number of the test person and the actual position.

Before starting the fault tolerant algorithm, the data has to be pre-processed. Therefore, the data of the movement sensors has to be loaded. Since the raw data offers the symbol "1" in case of an activation, additional "0"s have to be inserted (see Algorithm 6.10). The gained data is also aligned in respect with their time stamp as described in Algorithm 6.11. The data (*sensorData*) contains all sensor activation states at each time instance.

The next algorithm shows the course steps to determine incorrect activations. First, the counter number of incorrect activations (*incorrCounter*) is initialized. The counter stores the number of sequential incorrect activations. Active sensors are checked every iteration. If there are any active sensors, the reachable sensors are evaluated. One reachable sensor is directly applied to the person model. More sensors have to be treated as discussed in the scenarios depicted in Figures 5.9 to 5.11. Therefore, the function *getlinkedSensors* as presented in Algorithm 6.19 differentiates between these scenarios. It could be the case that no sensor is reachable after executing lines 5 to 10. This situation is indicative of incorrect sensor measurements. Moreover, the counter *incorrCounter* is incremented. When many sequential incorrect activations are observed, the person model is most likely stuck in an area. Thus, the person model is reset to the current position, obtained from *activeSensor* (line 17 to 19). The implementation of how to determine incorrect activations are provided in Algorithm 6.17.

Algorithm 6.17: determination of incorrect activations

```
incorrCounter = 0 // counter for the number of incorrect activations
for all sensorData do
  if sensor activations available then
    store these sensors in activeSensors
    activeReachableSensors = get all reachable sensors using isReachableSensor()
    if length(activeReachableSensors) ≥ 2 then
      activeLinkedSensors = getlinkedSensors()
    else
      activeLinkedSensors = activeReachableSensors
    end if
    if not isempty(activeReachableSensors) then
      incorrectCounter = 0
      personModel = overrideSensorsOfPerson(personModel, activeLinkedSensors)
    else
      // an incorrect activation happened
      incorrectCounter = incorrectCounter + 1
      if incorrCounter ≥ maxIncorrectNum then
        // reset person model to the actual position
        personModel = overrideSensorsOfPerson(personModel, activeSensors)
        incorrectCounter = 0
      end if
    end if
  end if
end for
```

The reachable sensors are determined by using the topology matrix of the actual flat. In addition to the topology matrix, Algorithm 6.18 also requires the array of all movement sensors and the person model. The output is the Boolean expression *reachableExpr* which indicates if the position of the *activeSensors* can be reached from the current position. Line 1 of the pseudo-code returns sensors which are stored in the person model. Next, the indices of *activeSensors* and *personSensors* in *sensorArray* are gained. The example of Figure 6.4 illustrates that these indices address the distinct entries of the topology matrix. The obtained *topologyEnties* indicate

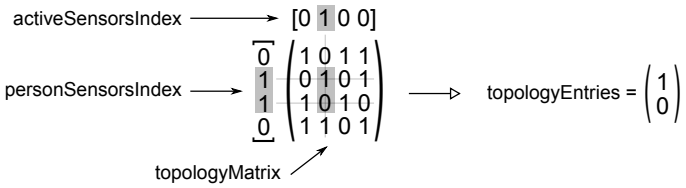


Figure 6.4: Demonstration of the result from function *isReachableSensor*

whether the desired desired sensor can be reached. In the example of Figure 6.4 one index of the *topologyEntries* is true. Therefore, the resulting *reachableExpr* is also true.

The implementation of the function *getlinkedSensors* of Algorithm 6.17 is given in Algorithm 6.19. The algorithm delivers the linked sensors of the active and reachable sensors called *activeReach-*

Algorithm 6.18: isReachableSensor()

Input: $topologyMatrix$ - topology of the flat, $sensorArray$ - array containing all movement sensors, $personModel$ - person model, $activeSensors$ - number of active sensors
Output: $reachableExpr$ - Boolean expression

1: $personSensors$ = get all sensors stored in $personModel$
2: $personSensorsIndex$ = get the indices of $personSensors$ in $sensorArray$
3: $activeSensorsIndex$ = get the indices of $activeSensors$ in $sensorArray$
4: $topologyEntires$ = get the entries of the $topologyMatrix$ at $personSensorsIndex$ (row) and $activeSensorsIndex$ (column)
5: **if** *topologyEnties* contains only zeros **then**
6: $reachableExpr$ = false
7: **else**
8: $reachableExpr$ = true
9: **end if**

ableSensors. If no links are present all sensors are returned. The links are those sensors that can be reached from one of the *activeReachableSensors*. Therefore, an adapted procedure such as *isReachableSensor* is used to get the links. The *linkedSensorsArray* stores all links. Duplicates may be present. If no links are found, all sensors of *activeReachableSensors* are returned. Present links have to be post-processed via the function *simplifyLinks*.

Algorithm 6.19: getlinkedSensors()

Input: $topologyMatrix$ - topology of the flat, $sensorArray$ - array containing all movement sensors, $activeReachableSensors$ - sensor numbers of the active reachable sensor
Output: $activeLinkedSensors$ - all active sensors that are linked

for all sensor combinations of $activeReachableSensors$ **do**
find links between $activeReachableSensors$
stored the found links in $linkedSensorsArray$
end for
if *not* $isempty(linkedSensorsArray)$ **then**
// sensors are linked
$activeLinkedSensors = simplifyLinks(linkedSensorsArray)$
else
// no links, so use all sensors
$activeLinkedSensors = activeReachableSensors$
end if

Algorithm 6.20 provides the pseudo-code of the function *simplifyLinks*. This function has to fulfill two tasks: the first is to merge equal links (line 2). The equal links of *linkedSensorsArray* are merged into an array of *simplifiedLinksArray*. Different links are stored at other indices of *simplifiedLinksArray*.

The second task is to elect the group(s) of links which include(s) the highest number of linked sensors (*maxLinksArray*). These links are combined (line 9 to 11) to one representation -

$simplifiedLinks$.

Algorithm 6.20: simplifyLinks()

Input: $linkedSensorsArray$ - array of all link combinations
Output: $simplifiedLinks$ - array of linked sensors without replicas

```
1: for all entries of linkedSensorsArray do
2:    find all equally linked sensors
3:    stored them into simplifiedLinksArray
4: end for
5: if isempty(simplifiedLinksArray) then
6:    // all linked pairs are different
7:    simplifiedLinksArray = linkedSensorsArray
8: end if
9: // the linked sensors with the highest number of links
10: maxLinksArray = get highest number of linked sensors from simplifiedLinksArray
11: simplifiedLinks = merge sensor list of maxLinksArray
```

7 Results

After having introduced the reader to algorithms and implementation details, the results are discussed. In case of the confidence weighted average algorithm and its extensions, different temperature profiles of an individual day are discussed. Furthermore, the behavior of the different extensions of the confidence weighted average algorithm is highlighted. Diagrams and performance analysis will indicate which algorithm is the most applicable. The second part discusses the outcome of the methods and algorithms to enable an easier configuration process of an Ambient Assisted Living system. Thus, the result of the node classification, topology learning, person model and their combination are reflected upon.

7.1 Raw data fusion

An important part of the thesis is fault tolerance of each individual method. If no fault tolerance is implemented, conditions that cause false positive alarms could occur as they would happen in systems without Sensor Fusion.

The confidence weighted average fusion algorithm and its extensions are discussed in this Section. The algorithm is used to combine temperature measurements (raw data) from different sensors. The fused measurements are gained from sensors that are close to each other. The first part will show the general behavior of the implemented algorithms. In a later step the fault tolerant behavior is shown. The used algorithm determines the impact of an erroneous measurement on the fused result. The following sections will highlight their individual advantages and drawbacks.

7.1.1 General behavior

The data of the investigation was produced by test person 2. One day was picked from the whole data set (2010-05-16) to show the different outputs. The individual day should be representative and should have a schema marked by transitions. The first algorithm of the confidence weighted average fusion gives the basic idea. In a further section the congeneric multi-sensor data fusion algorithm is presented. The other extensions are mentioned in the second part of the results, in which fault tolerance (see Section 7.1.2) is shown. They are not included in this Section because their outputs are the same in absence of sensor faults as the output of the confidence weighted average fusion.

Confidence weighted average fusion

In Figure 7.1 a typical temperature diagram is shown. The two temperature curves are from sensor 142 and sensor 147 (see Figure 2.1) which are about one meter apart. The standard deviation (±0.5℃) and furthermore the variances are the same for both sensors (see Section 2.2). In this case all sensors have the same influence (same weights) on the fused output (green curve).

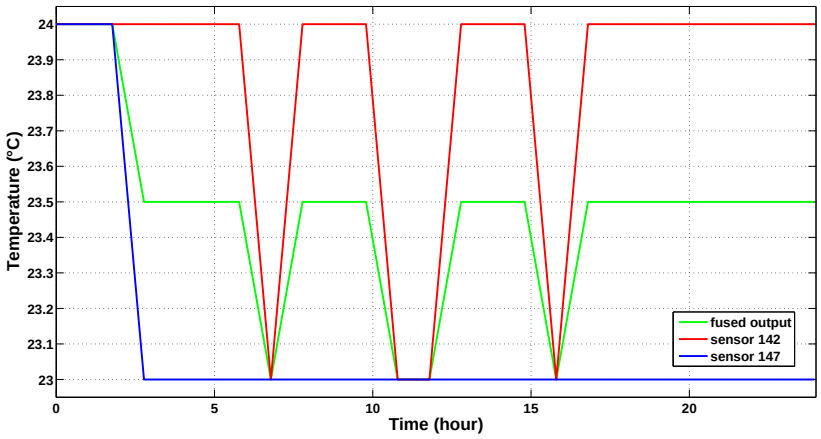


Figure 7.1: Confidence weighted average fusion

In order to express the significance of various sensor measurements on the fused result, different weights can be applied. These weights may express the locations of the sensors. The windows are not equipped with contact sensors. The measured temperature is the only indication of a window being open. When the temperature outside is lower than inside the flat, a nearby sensor

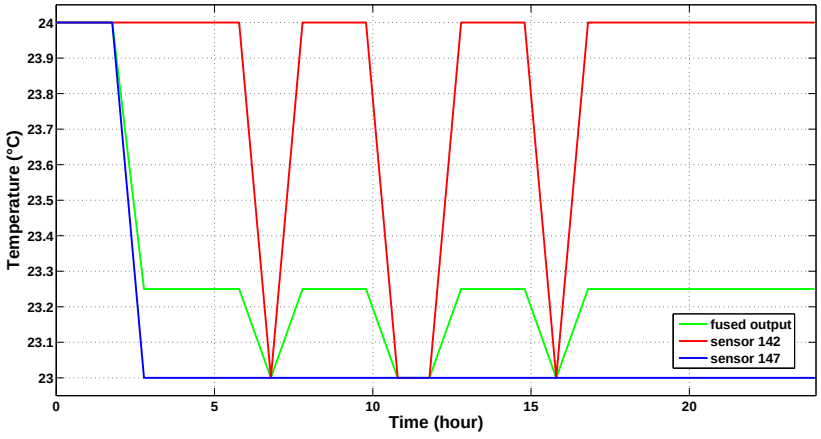


Figure 7.2: Confidence weighted average fusion with different variances ($\sigma^2_{147} = \frac{1}{3}\sigma^2_{142}$)

responds quicker to the temperature change. Furthermore, there will be a temperature gradient

which can be expressed by different weights of the measurements. If the focus is on observing an open window, more weight has to be assigned to sensors close to a window. If the temperature inside the room is more important for the observation, more weight has to be assigned to sensors with larger distance to a window. Sensor 147 is closer to the window than sensor 142. Figure 7.2 shows the same temperature schema for both sensors that are given in Figure 7.1. The fused output is different because the variance of sensor 147 is one third of the variance of sensor 142 (focus on an open window). The weights are 0.75 and 0.25 respectively. Therefore, the fused output curve shifts more towards the curve of sensor 147.

Congeneric multi-sensor data fusion algorithm

As discussed in Chapter 5, this extension of the confidence weighted average fusion algorithm re-calculates the variances for each measurement point. In Figure 7.3 the fused output for attenuation factor $\alpha = 0.5$ can be seen. The output is different from the one in the previous figures.

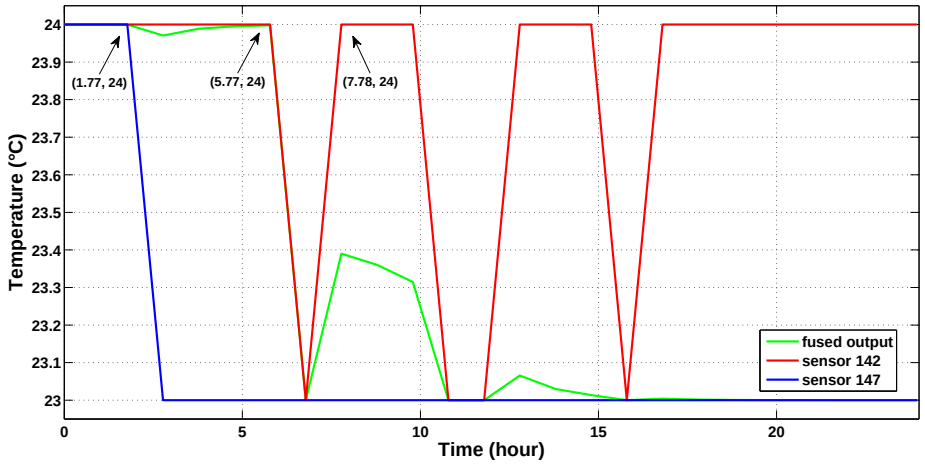


Figure 7.3: Congeneric algorithm ($\alpha = 0.5$)

In the first moment it is hard to grasp how the fused output is produced. Therefore, Figure 7.4 also includes the variances and the calculated weights. Before 1:45 a.m. (1.77), both sensors offer the same data value. After 1:45 a.m. sensor 147 changes to 23℃. According to Equation 5.7 the variance of sensor 147 increases because of the deviation of the current measurement value to the last fused value. The measurement of sensor 142 has less deviation from the last fused value. Thus, the variance is lower. After 5:45 a.m. (5.77) the variance of sensor 142 increases because of the changing measurement. At about 7:45 a.m. (7.78) the sensor 147 becomes more dominant which is also confirmed by its weight. In further measurements sensor 142 is weighted less so that the other fluctuations in the measurements are attenuated.

The algorithm represses measurements that differ from the last fused output. If more measurements confirm each other, their weights will also be in the same range. The time it takes to have equivalent weights is determined by the attenuation factor. The outputs, using different attenuation factors, are presented in Figure 7.5. The factors 0.1, 0.5, and 0.9 are used to demonstrate their impact on the fused result. An attenuation factor of 1 would not change the variances. They would remain the same in each step. In this situation the algorithm produces the same

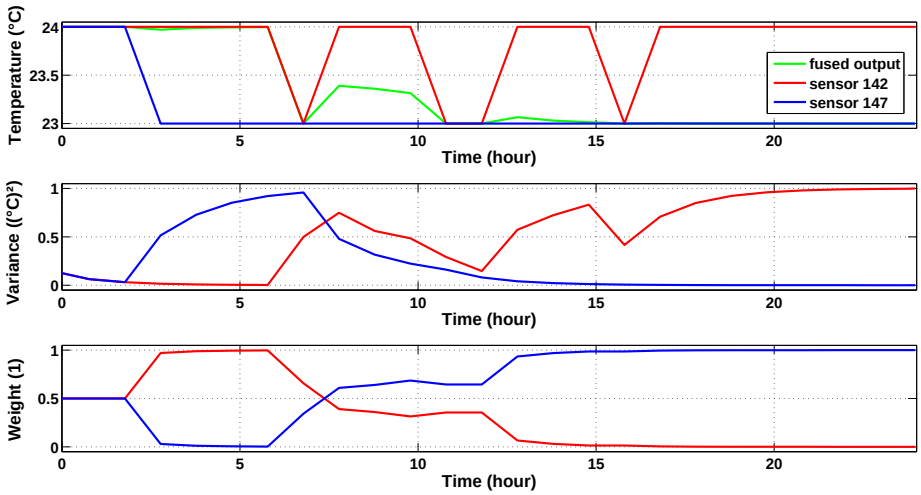


Figure 7.4: Congeneric algorithm ($\alpha = 0.5$) with variances and weights

output as the confidence weighted average fusion algorithm. When using high attenuation fac-

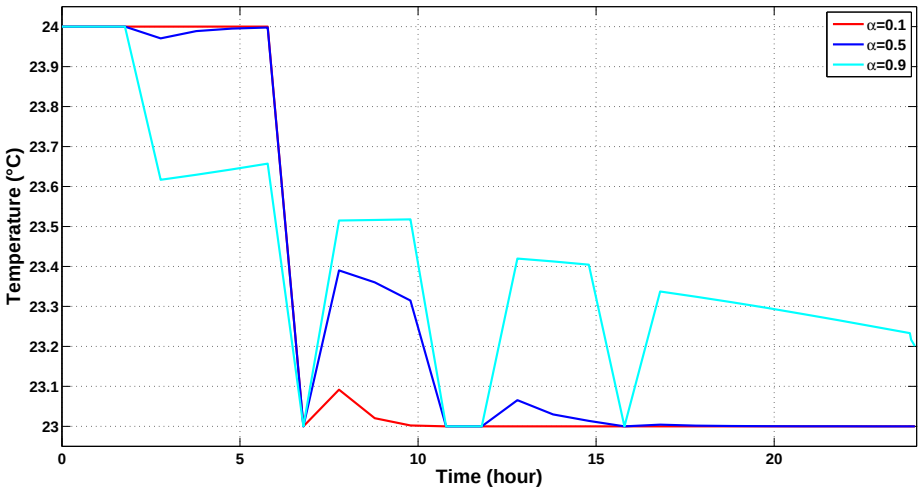


Figure 7.5: Output of the congeneric algorithm with different attenuation factors α

tors, changing measurements will have higher impact on the fused output than low attenuation factors. Low attenuation factors favor the recent measurement history which is indicated by the second term of Equation 5.7. The situation is indicated by the red curve ($\alpha = 0.1$) of Figure 7.5. It is evident that only the sensor measurement with minimal difference to the previous fused value is dominant. Higher attenuation factors (0.5 and 0.9) will have the effect that the previous variance is paid more attention to.

The appropriate factor cannot be given at this time. Moreover, the fault tolerant behavior (see Section 7.1.2) has to be studied first. After the following Section the proper attenuation factor will be given.

7.1.2 Fault tolerant behavior

The fault tolerant behavior of the individual algorithms will be discussed in order to highlight differences among the algorithms. Additionally, the adaptive algorithm and voting algorithm are presented. A sensor fault was never recognized by analyzing the data. Therefore, the data of sensor 148 was manipulated to illustrate the fault tolerant behavior. A set of sequential measurements with low and high values are inserted. The other sensor values remain the same as in the figures of the previous Section.

Confidence weighted average fusion

The impact of the erroneous sensor is presented in Figure 7.6. Every sensor is weighted equally with $\frac{1}{3}$. The fused output shows that the faulty sensor has a considerable impact. The first fault produces an output of about 15°C, while the second fault generates an output of about 27°C at about 3:00 p.m.

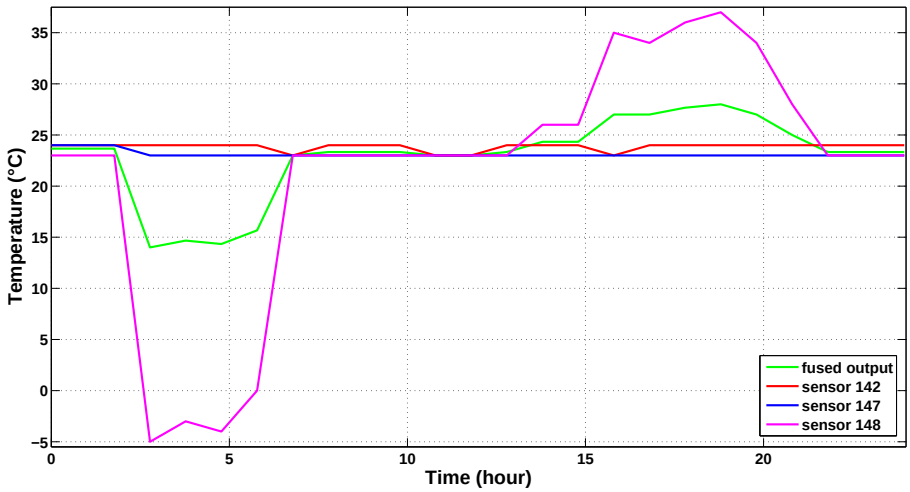


Figure 7.6: Confidence weighted average fusion using one faulty sensor (number 148)

Congeneric multi-sensor data fusion algorithm

The behavior of the algorithm has already been demonstrated in the previous Section. Figure 7.7 highlights the diagrams of the temperatures, variances, and weights of each sensor. The temperature diagram only shows the faulty sensor and the fused output. Initially, all sensor measurements are weighted equally. After the occurrence of the first fault, the variance of the affected sensor increases dramatically so that the weight of this sensor is about zero. After recovering from the sensor fault, the variance is decreasing. It takes a certain amount of time which is determined by the numbers of samples taken in the interval and the attenuation factor. The sensor value itself is not in close range to the fused output. Thus, the weight does not get high enough to contribute considerable to the fused output. In the second erroneous measurement

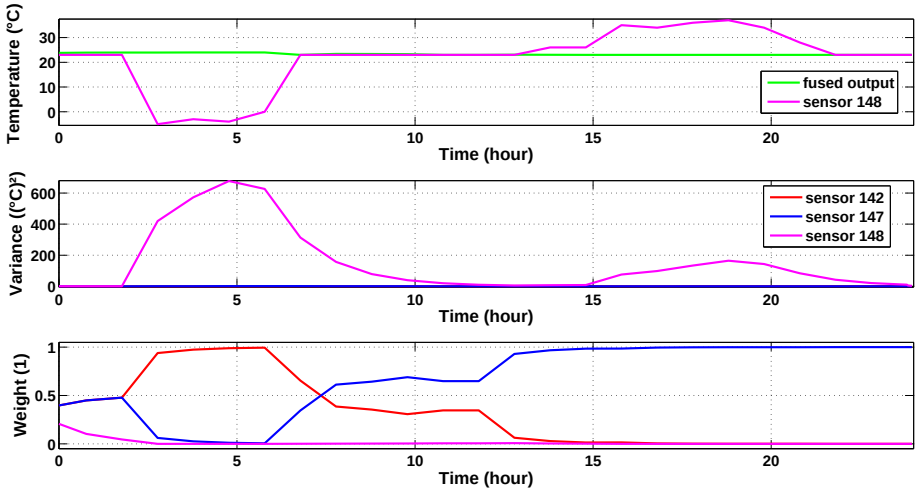


Figure 7.7: Congeneric algorithm ($\alpha = 0.5$) with faulty sensor 148

period the variance is increasing again. However, the variance is not reaching the same values as at the first fault because the deviation from the fused output value is lower.

The graphs of the correct sensors mainly follow the same schema as those in Figure 7.4. It is also true for the variances that cannot be given exactly. The variance of the faulty sensor is dominant. The schemas of the weights display a similar behavior. Using different attenuation factors, Figure

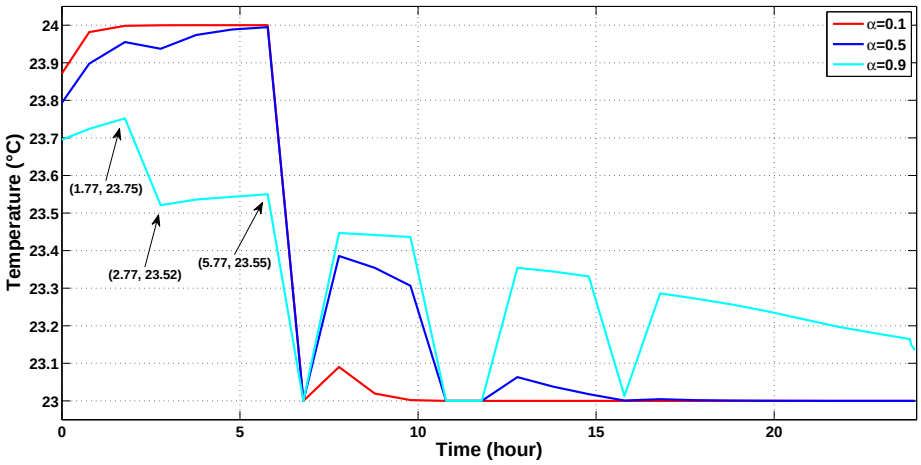


Figure 7.8: Output of the congeneric algorithm with different attenuation factors α

7.8 offers similar results compared to Figure 7.5. The main difference is that the faulty sensor yields some contribution to the fused output in the time interval from about 2:45 a.m. until 5:45 a.m.

In the last section the question of the optimal attenuation factor was raised. The observations of this Section show that a faulty sensor is weighted less so that it cannot contribute significantly

to the fused output. After recovering from a fault, it takes a certain time to be able to influence the result again. This behavior has already been discussed in the last Section. An attenuation factor in the rage of 0.4 to 0.6 turned out to be practical.

Voting algorithm

The modifiable parameter in this algorithm is the distance parameter. In [DLC05] a distance parameter of $\pm 2\sigma$ or $\pm 3\sigma$ is recommended. Within these fault conditions a distance parameter of $\pm 2\sigma$ (± 1°C) has been selected.

At this point the sensor behavior should also be considered. The standard deviation σ of a temperature sensor is ± 0.5°Cbut a sensor just transmits the values with a resolution of 1°C, which equals to two standard deviations. The enlargement of the distance parameter to $\pm 4\sigma$ should be thus considered.

Figure 7.9 displays the faulty sensor 148 and the fused output. The voting scheme excludes sensor 148 if it cannot collect the majority of votes. This is also true for the two intervals with the incorrect measurements. The constant temperature of 26°C from about 1:30 p.m. until about 3:00 p.m. doesn't appear in the fused output.

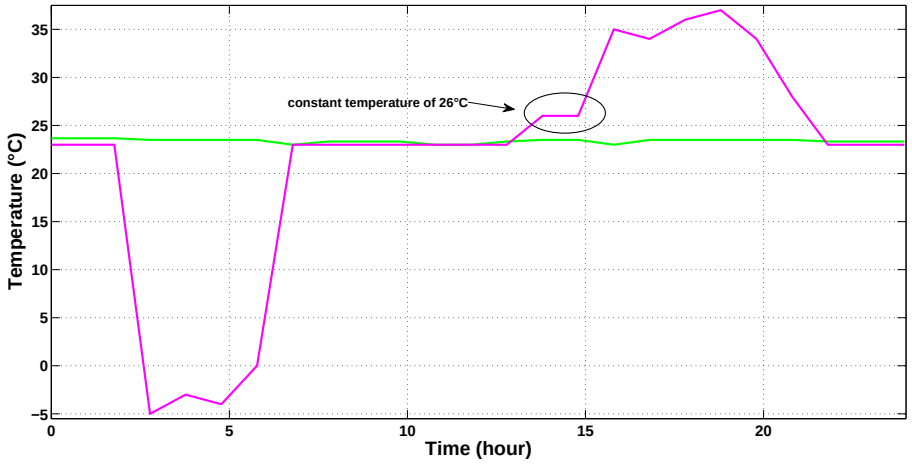


Figure 7.9: Output of the voting algorithm ($d = \pm 2\sigma$)

Adaptive algorithm

Equal distance parameters applied to the voting algorithm and the adaptive algorithm produce the same outputs. This situation was observed with regard to the applied sensor fault. However, it is not true in general. Figure 7.10 shows a different result compared to the voting algorithm when the distance parameter is set to $\pm 4\sigma$. The difference to Figure 7.9 is the slightly increased value before the second fault is visible in the fused output.

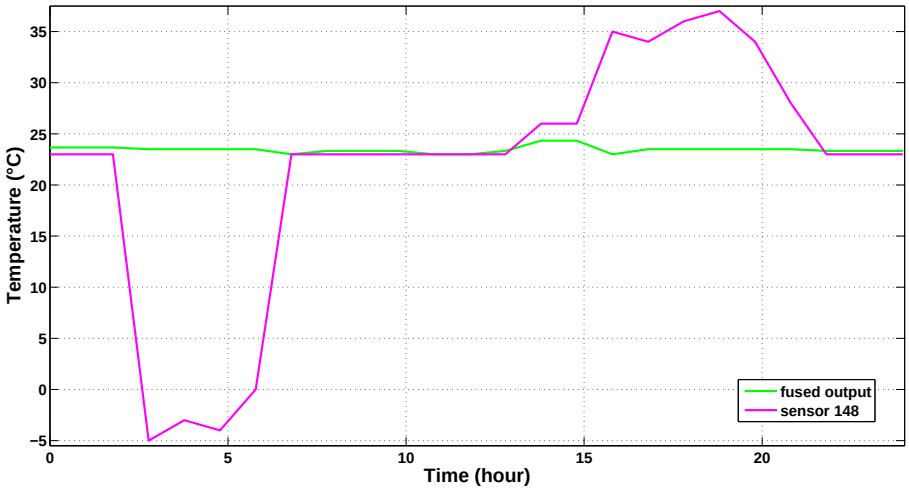


Figure 7.10: Output of the adaptive algorithm ($d = \pm 4\sigma$)

7.1.3 Comparison of the algorithms

The first step is to compare all algorithms that have been applied and to find a subset that suits this task best. Figure 7.11 compares the fused outputs. The curves of the voting and the adaptive algorithm overlap except in the short period before 3 p.m. Moreover, the congeneric algorithm

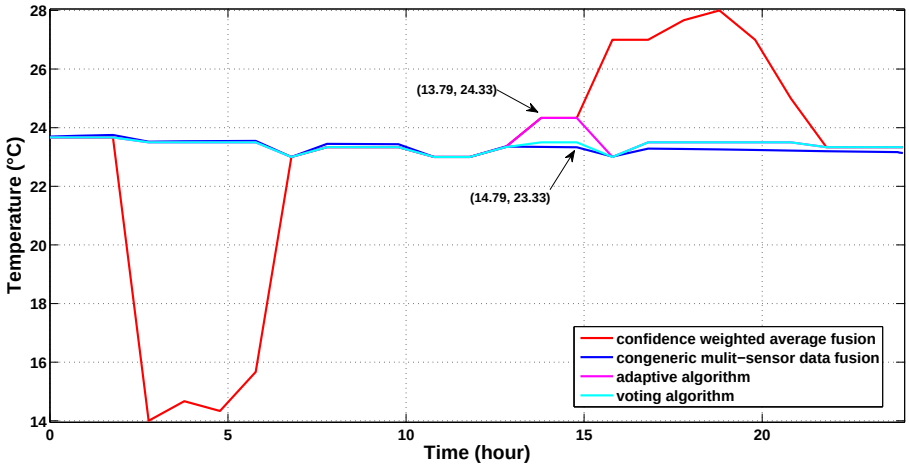


Figure 7.11: Comparison of the used algorithms

fuses the input values that produce similar outputs as the adaptive and the voting algorithm. It is clearly evident that the confidence average weighted fusion algorithm is affected by the incorrect sensor. In order to highlight the impact of the erroneous measurements, the fused temperature is compared to an upper and to a lower threshold. This concept is used to determine an open window (too low temperature) or a heat source (too high temperature).

Figure 7.12 shows that these thresholds are exceeded twice at that day. In the time interval

from about 2:46 a.m. (2.77) until 5:45 a.m. (5.77) the fused temperature is below the low temperature threshold. In the afternoon the high temperature limit is exceeded. Whenever a limit

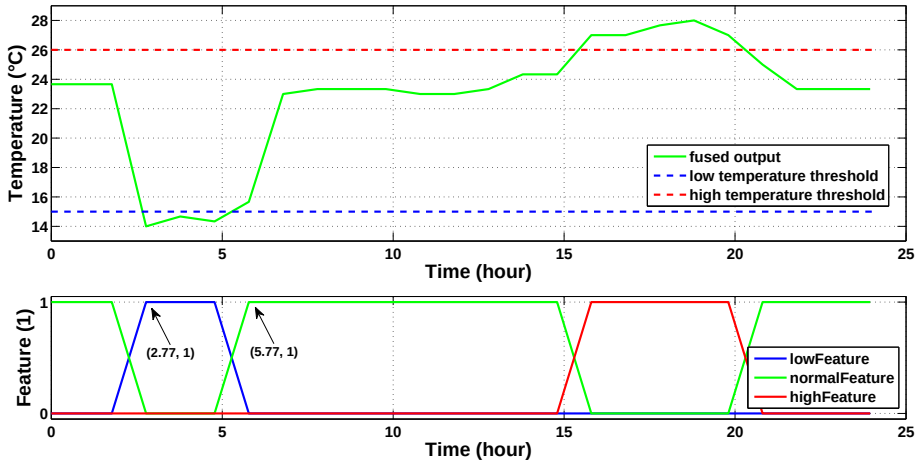


Figure 7.12: Output and features of the confidence weighted average fusion algorithm

is exceeded, a feature is generated. The second sub-figure shows that the features *lowFeature*, *normalFeature*, and *highFeature* are set to true when the temperature is within its specified bounds. Recommendations for the maximum and minimum temperatures are given in [16]. As a consequence, the lower threshold is set to 15℃ and the upper limit to 26℃. When the temperature is within the range of 15℃ and 26℃ the *normalFeature* is set to true. The created features from Figure 7.12 do not reflect the actual condition. Moreover, these features would cause false positive alarms. The created feature of the other algorithms would always be the *normalFeature*.

A decision between the three remaining algorithms has to be made. The performance has to be analyzed in terms of e.g. number of iterations and computational complexity. The following table (Table 7.1) gives an overview of the number of iterations needed to produce a fused output and the number of sensors to maintain the fault tolerant operation. The congeneric algorithm just

Table 7.1: Comparison of the fault tolerant algorithms

Algorithm	Iterations	Number of sensors
Congeneric	1	≥ 2
Voting	$n(n-1)$	$2k+1$
Adaptive	$k+1$	$2k+1$

needs one iteration to produce one output. In the single iteration the new variances are calculated and the confidence weighted average fusion algorithm is used. At least two sensors are needed to preserve a fault tolerant behavior.

Using the voting algorithm, $n(n-1)$ iterations are needed to determine the correct sensor measurements. Therefore, n sensor measurements are available and each of them has to collect votes from the remaining $(n-1)$ measurements. After collecting the votes, the confidence weighted

average fusion has to be executed once. The number of measurements in correspondence with the number of sensors has to be $2k + 1$, k being the number of the sensor faults.

The adaptive algorithm also needs $2k + 1$ sensors, so that the majority is facing no fault. When no fault occurs just one iteration is needed. The number of wrong sensor measurements k causes another k iterations. In each iteration one faulty sensor is excluded and the confidence weighted average fusion is computed.

Finally, one can conclude that the congeneric multi-sensor data fusion algorithm is the most appropriate one for countering this problem. The variances of the sensors do not have to be priorly known. The variance has only to be set initially (see Section 6.1.2). Further iterations re-calculate the variance. It is best practice to set all initial variances to the same value. The initial fused output value is set to the confidence weighted average of the first sensor values. This value can also be set to another value that is within the data interval of the sensors. The initial fused value will have less impact on the algorithm because similar sensor values will produce approximately the same weights. This will produce a fused value that is in the range of these sensors measurements.

Another reason for choosing this algorithm is that the majority has not to be met. The outliers are restrained automatically by their variances. The sensor density of an area is often low. In this scenario just two sensors are frequently present for the fusion process (see Figure 2.1). The voting algorithm needs the majority of the votes which would be a minimum of three to tolerate a single sensor fault.

When the sensors are far enough apart, the distance to a cooling or heating source can be modelled with different weights as mentioned in Section 7.1.1. Applying higher weights to faulty measurements could result in the fused output having more deviation from the correct sensor measurements than from the incorrect in the adaptive algorithm. It is possible that only the incorrect measurements produce the output after the iteration process is finished. Thus, it would be better to apply the voting algorithm. First, the erroneous sensor measurements are excluded, followed by the execution of the confidence weighted average algorithm. The weights have no influence on the selection of the neglected sensors.

Despite the facts stated above, the preferable algorithm is still the congeneric multi-sensor data fusion algorithm. Spatial deviations in the sensor placements can be counterbalanced by the applied minimum sending interval of one minute (see Section 2.2). Another advantage of this algorithm is that the configuration work is also minimized when no explicit assignment of variances has to be done.

7.2 Node classification

Performance criteria are evaluated to give evidence to the performance of the classification process. The performance can be described in terms of e.g. classification rate and the time it takes to identify all sensor types. It is also investigated which sensor nodes and sensor types are often misclassified. The distinct number of unclassified sensors is also given.

In order to deliver a representative picture about the algorithm's performance, the classification was done at each day of the recording interval. The data of all three test persons (TP2, TP3 and TP4) are scrutinized in order to highlight differences.

The next Figures (7.13 to 7.16) show how many of the overall sensor types can be identified on a single day. The value of the recognized types is always in relation to the total number of sensors.

The data interval of test person 2 had to be split because sensor node 155 had been replaced by sensor node 184 after appearance of a hardware fault. Figure 7.13 depicts the first part of the recording period. On average, three sensors are not identified (excluding the highlighted days of vacation and insufficient diary records). The diary records are fragmentary, thus offering no explanations for the actions of the first period. It can be assumed that the person was not all the time in the flat. The test person was on vacation in the second period. That is also the reason

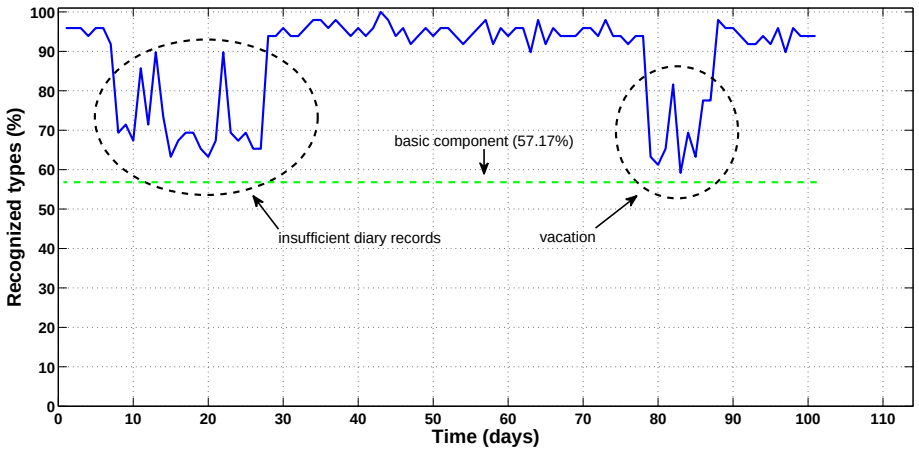


Figure 7.13: Recognized sensor types in the flat of TP2 (first time interval)

why fewer sensors have been identified. The basic component delivered from temperature sensors and light sensors is 57.17 percent of all recognized sensors. In contrast to temperature sensors and light sensors, the other sensor types show an event-triggered behavior (see Section 2.2). They just provide measurements when an event happens. The deviation from the basic components can be explained in terms of faulty sensors.

The second half of the recording period, including node 184, is depicted in Figure 7.14. The basic component is 56 percent. The highest value of unrecognized sensors is five. On the last day all installed sensors are recognized. On average, two to three sensors cannot be recognized.

In the first part of the recording period the door contacts of the nodes 152, 153, 143, 148 were unrecognised most often. At some days also the accelerometer and the movement sensor of node 148 could not be recognized. The second interval shows similar behavior: the same sensors could not be identified.

The data from test person 3 in Figure 7.15 delivers one interval with lower recognition rates when the person was on vacation. In the remaining time interval two sensors could not be recognized on average. The green dashed line indicates the basic component of observable sensors if no activity is present. The basic component of 52.63 percent is identified on day 46.

In most cases the door contacts of node 105, 106, 107 and 108 have not been identified. Similarly, the movement sensor of node 112 is not identified on 11 days. The included accelerometer noticed some activity. The mounting on top of the toilet's door frame is not practical. Causes for the disturbance of this sensor can be e.g. hardware fault, or barrier in front of the sensor.

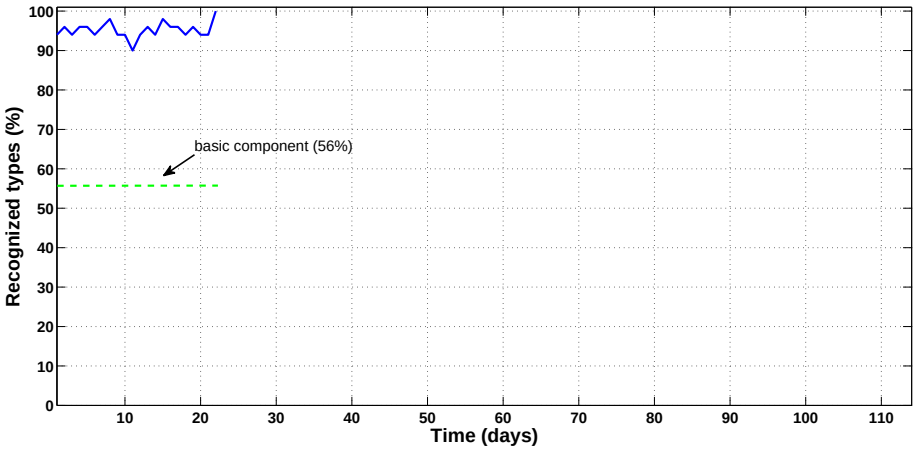


Figure 7.14: Recognized sensor types in the flat of TP2 (second time interval)

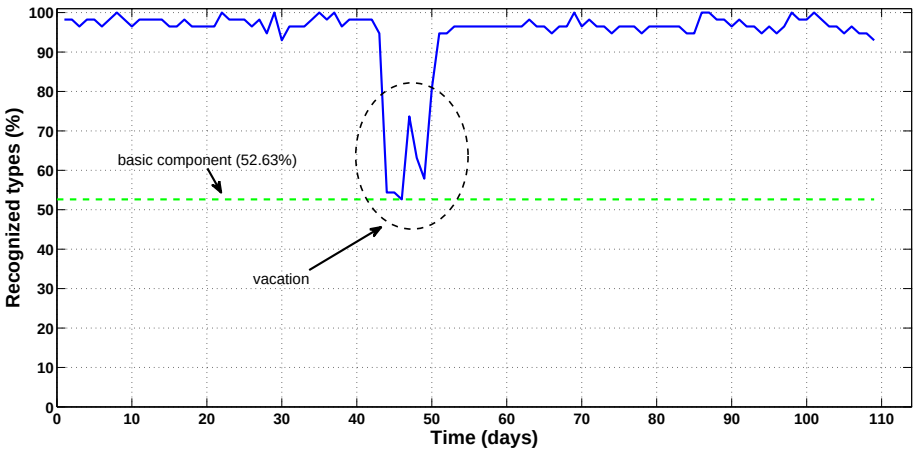


Figure 7.15: Recognized sensor types in the flat of TP3

The node classification of test person 4 shows a basic component of 57.78 percent. On average three sensors cannot be recognized in the recording interval.

Figure 7.16 shows that one sensor is not identified on average. Similar data is offered with regard to the other flats. The door contacts could also not be recognized on each day. On some days also the accelerometers did not deliver any data. In this flat the nodes 123, 126, 127, 133, 134, and 136 are nodes that include these sensors.

The observations showed that the door contacts could not be recognized in most cases. This is due to the fact some doors stay in the same state for a longer time. Some accelerometers and movement sensors also show a disadvantageous placement of the sensor node. Especially the placement of node 112 should be revised. As an additional benefit, the node classification process highlights the shortcomings in the placement of the nodes.

Sensor faults can only be partially evaluated. A faulty sensor, bad sensor placement or the absence

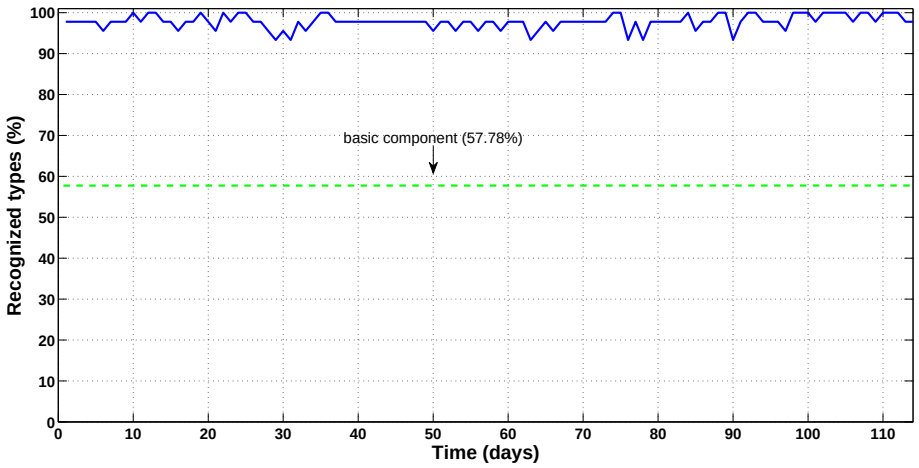


Figure 7.16: Recognized sensor types in the flat of TP4

of the person can be reasons for event-triggered sensors not providing any data. The differentiation between these three conditions cannot be done in regard with the node classification process. In contrast to this, the lack of temperature and light sensor data would indicate a hardware fault so that these types would not be recognized any more. This would imply a fail-silent[1] behavior of the node. Such a condition is never observed but could be a reliable indication of an erroneous sensor.

Another point that has to be addressed is the behavior of the classification process in case of the test person's absence (vacation). The smaller peaks in the periods in Figure 7.13 and 7.15 are caused by faulty sensor measurements. Incorrect sensor data are caused by solar radiation, too high sensitivity, hardware faults, and other environmental influences. The diary and other protocols give no explanation for the higher peaks. Obviously some kind of maintenance must have been carried out.

The next statistics will give the amount of incorrect classifications on the single days. The percentage is calculated on basis of the incorrect classifications in comparison to the total amount of sensors. In the first interval the data (see Figure 7.17) from test person 2 shows that two sensors are incorrectly classified on average. The value of 2.04 percent is equivalent to a single sensor fault. In the second interval only one sensor type is not classified correctly after replacing node 155.

The classification results of the other two flats are even more unambiguous. All sensors of these two flats can be classified correctly. These classifications were done according to the data intervals for the individual sensor types. The intervals have been gained from observing the whole data interval of each data set and combining them to a consistent interval. The intervals are adjusted to cover a broader range of conditions.

Next, the questions of how long it takes to identify all sensor types and what improvements are achievable need to be addressed. The common way of installing the nodes and letting them gather sensor values is described in Table 7.2. On each additional day more information can be collected

[1] fail-silent - the faulty node would reject from the communication medium, i.e. it would not send any further sensor data

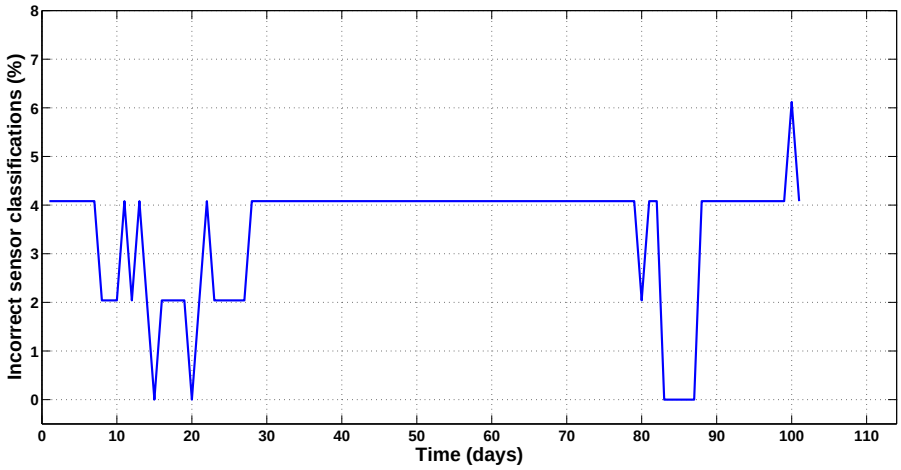


Figure 7.17: Incorrect classified sensor types of TP2

Table 7.2: Node classification statistics

Test person number	**Recognition rate after first day**	**Time until all sensors are identified (days)**	**Total duration (days)**
2	47/49	43	124
3	56/57	6	109
4	44/45	10	115

for the classification process. All sensors can be identified after a specific time interval. The figures show that just one or two sensors cannot be identified after one day. It takes additional 42, 5 and 9 days respectively to tag the absent sensors.

The previous figures and tables demonstrate that the sensor classification can be done at high rates. The maximum number of three sensors cannot be classified on a single day (see Figure 7.17).

In the classification process itself overlapping intervals are problematic. The classifications focus on finding non-overlapping data intervals. The light sensors and accelerometer are good examples. When an accelerometer observes no major vibrations, the sensor values lie in the data interval of a light sensor. Thus, an accelerometer could be classified as a light sensor. Usually, the accelerometers offer maximal sensor values that are up to five times higher than the highest values of the light sensors. The overlapping intervals occur when an insufficient amount of measurements is available.

When a single node classifies its sensor types, the main drawback is the time it takes to do this. The learning time depends on the occurrence of events. Events are rarely produced by reed contacts. Inside a flat some doors remain in the same condition. The rare usage extends the learning time.

The problem is tackled by exchanging the classification information among the node. Therefore, it is not necessary that each sensor node has classified all of its included sensor types. Moreover, it is necessary to know which sub-type numbers (see Section 2.3) it sends. The node is sending a unique sub-type number to a central node, not knowing what the number stands for. The numbers have to be identical in each node. After making these assumptions each node classifies its sensors and shares the result with the other nodes. Using a majority decision the type indicated by the sub-type number is determined. The combined decision is communicated to the other nodes, so that wrongly classified types or even missing classifications are corrected.

The high classification rate of a single node helps to correctly classify each node. The broader view of all classifications makes it possible to reduce the learning time to just one day. In comparison to the individual learning times of TP2, TP3 and TP4 these intervals would be 43, 6 and 10 days respectively. It shows that this kind of fusion (*decision fusion*) improves the performance of the system significantly.

7.3 Topology learning

The performance of the topology learning algorithm is investigated. It is necessary to discuss all influences on the sensor data and in later step on the algorithm itself. The learning interval also needs to be discussed.

The first part addresses the assumptions regarding the implemented algorithm. Individual topologies will be discussed in greater detail. The influences on the learning process will be highlighted.

In course of the implementation in Algorithm 6.9 of Section 6.2.2 it was mentioned that only the transitions between the sensors are necessary. The simple reason is that continuous data from just one sensor would produce an increased number of the events in the secondary diagonal of the compatibility matrices. The secondary diagonals are used to get the indication of uncorrelated sensor pairs. The increased number of events would lead to a greater number of uncorrelated sensors and would also require setting another threshold for the classification process. The unprocessed data would not produce a representative picture of the real topology. No distinction between sleeping or every day activity has to be made when using the transitions only.

The other issue is the spatial arrangement of the sensor nodes. Moreover, the arrangement of the nodes and the distances between them define the time it takes a person to get from one coverage area of one sensor to another one. In the implementation the time interval is a tuning parameter. If the parameter is set too high, many nodes will be linked together. On the other hand a too low time interval will produce less links. That could also result in to the problem that evident relations are not found. A good balance between these situations is found when the time interval is set in the range of 3.5 to 5 seconds.

The next figures show the topologies of the three flats. The graphics just show the nodes that include movement sensors. The dashed lines depict the links between the sensor nodes.

Figure 7.18 presents the topology of the flat of test person 2. The sensors in the corridor and in the bedroom form the main sensor links. Every other room (node) can be reached from these sensors. The movement sensor in the bedroom faces towards the corridor and is able to detect events outside the bedroom. Without any knowledge about the activations of the contact sensors it can be concluded that the bed room door is open most of the time. At the first learning attempt a relationship between sensor 145 in the living room and sensor 148 of the bedroom

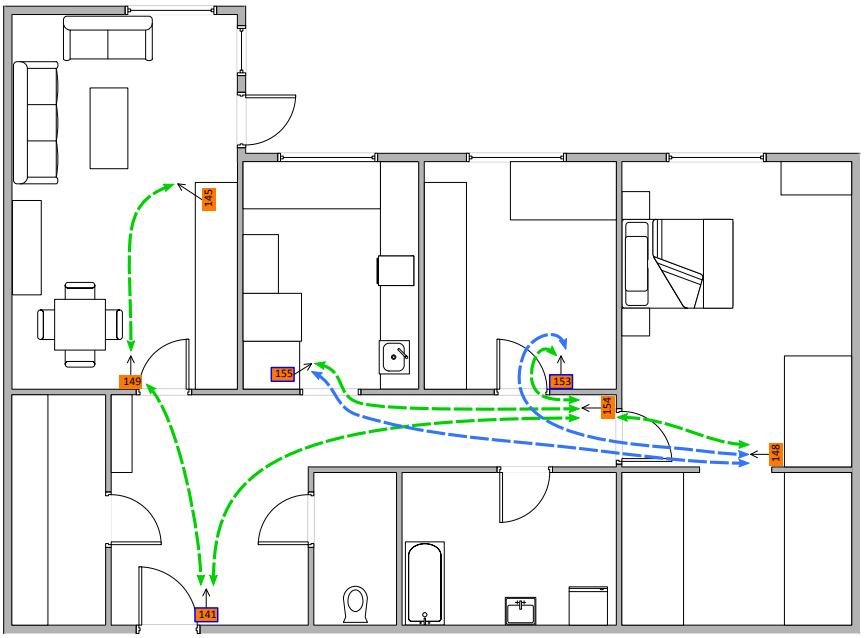


Figure 7.18: Flat topology of test person 2

became evident. According to the room geometry both sensors are totally disjointed. So the diary gave evidence that sensor 145 had many false positive activations. The correlation is a side effect of incorrect sensor data due to solar radiation. This problem is solved by using the data only when the sun cannot disturb the measurements. The same procedure is used for the other flats.

The flat of test person 3 is larger. A higher number of nodes are installed to cover the whole area. Figure 7.19 depicts the placement of the movement sensors in flat 3 while the topology is illustrated by the graph of Figure 7.20. A difference between the real placement and the outcome of the learnt topology is observed.

Especially sensor node 108 (bedroom) shows more links that should not be present. The connections to node 110 and 111 are not evident and should be handled with care. The real sensor placement shows no distinct connection between the rooms of installation. The use of the data at different times of the day did not bring any improvement. Things got even worse because more links had been created. Figure 7.20 shows the best outcome of the learning process.

Another outcome is that sensors that are in close proximity to each other might not even be linked in the learnt topology. This conclusion can be drawn from the sensor groups 105, 106, and 107. The first two nodes are very close to each other but are observing different directions. Sensor 107 observes the area towards sensor 105. This creates the links between these nodes. The same is true for node 106 and 107. The probability that close sensors facing the same direction are linked is greater than if facing different directions.

The simplest topology belongs to TP4 (see Figure 7.21). The topology is formed by a connected path, beginning at the entrance door (lower left corner) and ending in the kitchen (upper right corner). Originally, one additional sensor was present in the corridor: in close proximity to sensor node 127. The node did not provide clear data. Thus, the sensor could not be used at all. Sensor

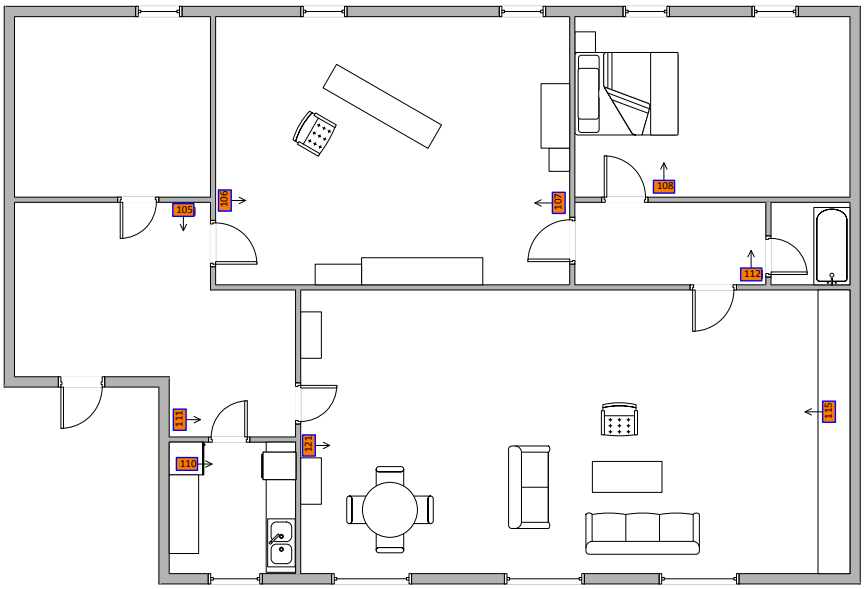


Figure 7.19: Room topology of test person 3

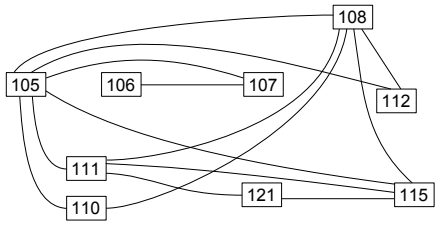


Figure 7.20: Sensor links in the flat of test person 3; best result

133 in the kitchen also was exposed to interference due to solar radiation. The learning intervals had to be limited to the morning hours.

Another result of this investigation is that too many links can indicate that the sensor density is too high. This can only be confirmed if the exact geometry of a flat is known.

Finally, the reasons that make it complicate to identify the topology are given in the following:

- Hardware faults (broken sensors, blinded sensors)
- Incorrect events (e.g. solar radiation)
- Sensor density
- Visitors
- Pets

Hardware faults and false positive events create relationships between nodes that do not reflect the real situation. Inconsistent sensor readings are also observed when more persons or pets are present in the flat.

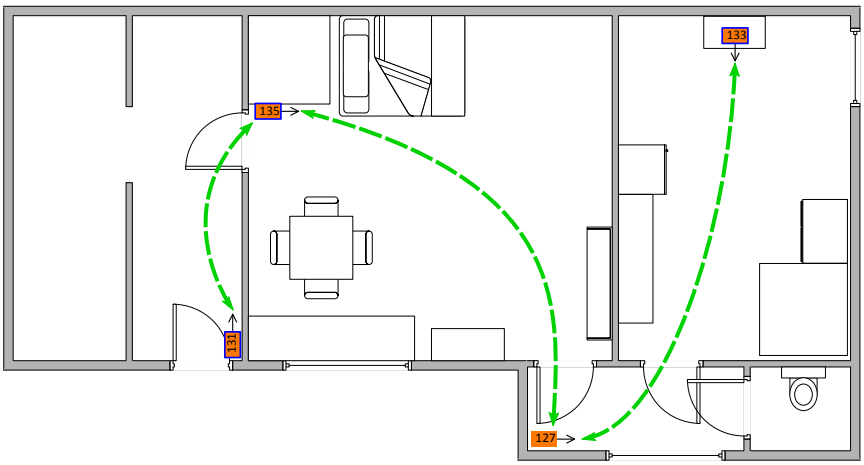


Figure 7.21: Room topology of test person 4

An appropriate learning interval compensates for rare negative impacts. The learning interval is between one day and ten days. The learning interval depends on the complexity of the room topology and the number of installed sensor nodes.

7.4 Combination of the methods

The results of the node classifications show that it is no problem to identify all movement sensors. These sensors are used to determine the topology. The results highlighted describe the combination of the topology and the person model.

No separate chapter is dedicated to the topic of the person model since it is no stand-alone method. The person model on its own only indicates the current position of a person.

Figure 7.22 depicts the sensor activations triggered by test person 4 in a random time interval. The activation states of four movement sensors are illustrated. Sensor 131 (entrance) and sensor

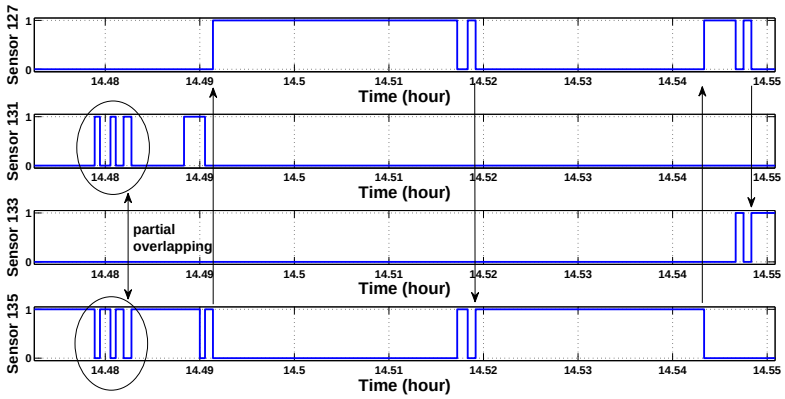


Figure 7.22: Sensor activations of test person 4

135 (living room) were activated first. When the person is in the coverage area of both sensors, the diagrams show frequent transitions between these sensors. The next movements were observed in the corridor (sensor 127) coming from the living room and in the kitchen (sensors 133). In reference to the topology of Figure 7.21, the activations must not be corrected. They are always correct in this short period of time.

The following result shows some measures to estimate the performance and quality of the topology. Furthermore, the corrected sensor values are highlighted. The most frequently excluded sensors are presented. They are compared to the diary entries. Another question is how many sensors are assigned to the person model and what conclusions can be drawn from these observations.

First, the model application to test person 2 is discussed. The topology learning process turned out to deliver a good representation of the real sensor arrangement. Figure 7.23 shows different numbers of sensors assigned to the person model. In the morning the person model is reset 8

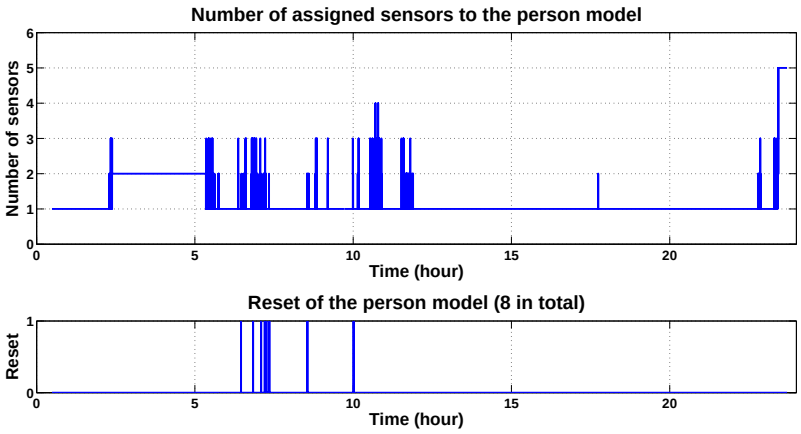


Figure 7.23: Temporal behavior of the flat of TP2

times. The limit for sequential incorrect activations is set to five. After five incorrect activations the person model is supposed to be correct again. The reset of the current position partially overcomes the problems of a corrupted topology.

The data of Figure 7.24 show how many activations of a sensor are corrected in total compared to the total number of events. According to the diaries sensor 145 had many incorrect activations. However, these statistics do not confirm the observation. Most activations had to be corrected in sensor 141 and 149. These are the sensors next to node 145. Further investigations on this issue showed that these corrections (exclusions) are caused at the transitions from sensor 141 to sensor 153 and 155 and also in the reverse direction. These are the sensors that are not related to each other. The transitions between these sensors had been observed too rare to be part of the topology.

The next Figures depict the investigation of the flat of test person 3. The topology is more complex and has more links between individual sensors. Figure 7.25 displays a higher number of active sensors. The topology contains a few linked sensor groups. Despite larger linked groups, 33 resets - which is a relatively large number - of the person model are observed. Therefore, the average number of sensors included in the person model is higher than in the previous example. Figure 7.26 shows the sensors producing the incorrect positions. The first assumption is that

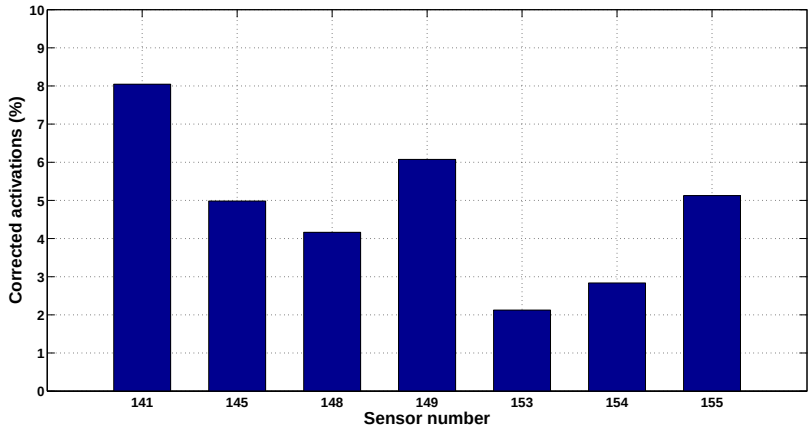


Figure 7.24: Corrected activations of flat 2

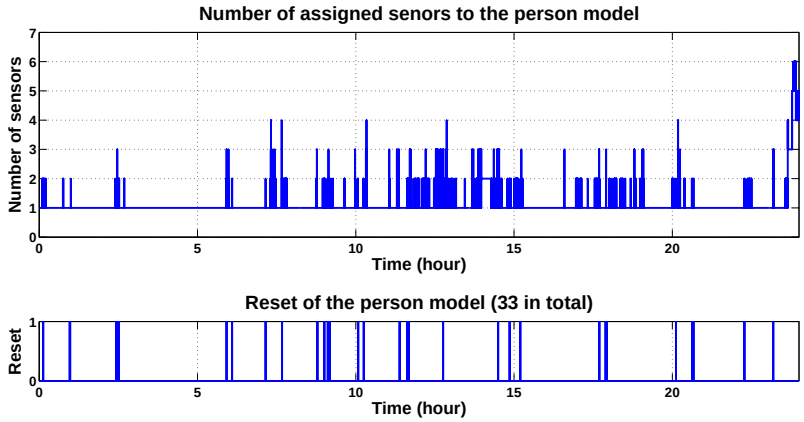


Figure 7.25: Temporal behavior of the flat of TP3

sensor 108 is linked incorrectly to sensors 110, 105 and 111. This should also be confirmed by the data. In fact, the linked nodes of sensor node 108 produce the incorrect activations. The sensor nodes 105 and 108 have a lot of options for their next position - they produce less incorrect events. The nodes that have only two connections produce more incorrect activations.

The topology in the flat of test person 4 is very simple. Thus, the best results should be obtained. According to the diary records, sensor 133 in the kitchen made quite a lot of incorrect measurements due to solar radiation.

At the random day of Figure 7.27, the average number of corrected active sensors (sensors assigned to the person model) is one. The highest value is three after waking up and in the evening.

The diary records are confirmed by Figure 7.28. The events of sensor 133 are corrected in about 2 percent of all data values. Sensors with many incorrect activations can be identified this way. It is necessary to find out if the faults are transient or permanent. The data and also the diaries confirm that sensor 133 faced the many incorrect events which are transient.

The first conclusion that can be drawn from the investigation above is that the whole correction

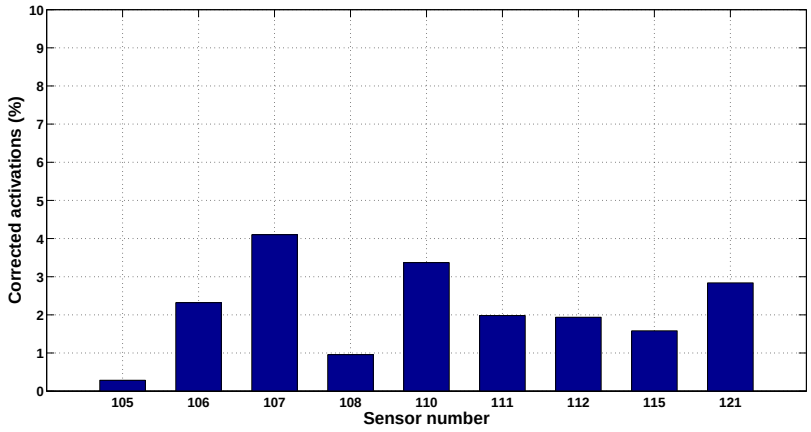


Figure 7.26: Corrected activations of flat 3

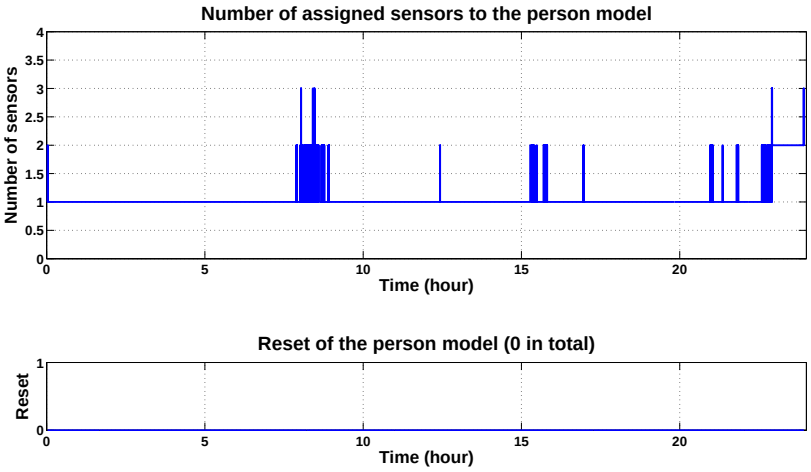


Figure 7.27: Temporal behavior of the flat of TP4

process depends on the topology. The topology's quality of flat 3 is not that "good" because a lot of reset actions happened. The number of active sensors assigned to the person indicates the number of sensor links and the sensor density.

The reset count is not always the appropriate indicator of a "good" or "bad" topology. The scenario of getting trapped in an area also produces reset values. This is caused by unfavorable activations when getting into the reachable area of an erroneous sensor.

In order to use a time series analysis such as hidden Markov models, the time is split into intervals (e.g. 15 minutes). The activation of a sensor is determined by integration or adding the times of activations. These sums are compared to the total area of this interval. If a certain ratio between these sums is exceeded, an activation of the sensor is supposed to be present in this interval. Missing activations can be compensated by lowering the bounds for an active interval.

The analysis showed that the combination with the accelerometers is not possible. The sensors do not show quite good enough information to fuse them with the movement sensors' data. It

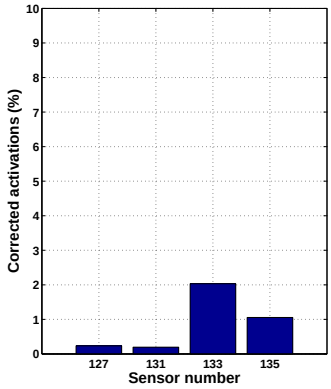


Figure 7.28: Corrected activations of flat 4

would be a better solution to decide between two paths like it has been demonstrated in Figure 5.9.

At the end of the days more sensors got assigned to the person model (see Figures 7.23, 7.25 and 7.27). More persons present will cause more activations and also more sensors will be included in the person model. The last analysis concerns the behavior of the combined model when more persons are present. Therefore, the next figures depict the number of sensors included in the person model and the diagram of the resets. The marked time intervals indicate the times when at least one additional person is present.

In the highlighted time intervals of Figure 7.29 the number of active sensors has increased. In addition more reset actions happened. In contrast to previous results, the reset counter depicted

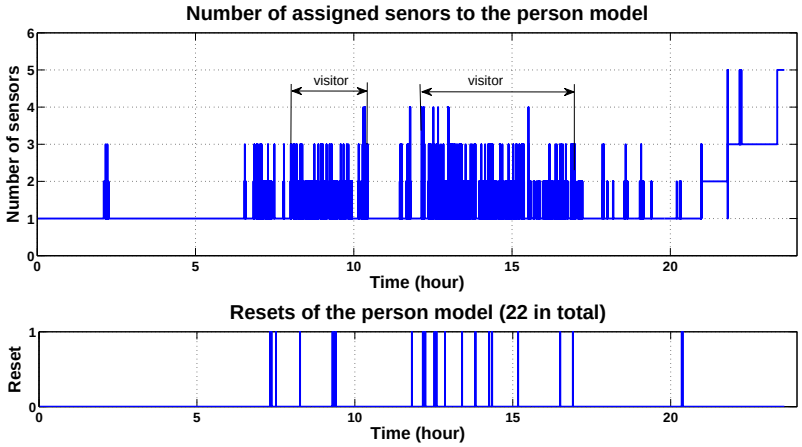


Figure 7.29: Temporal behavior in presence of a visitor (TP2)

in Figure 7.30 remains zero and the number of activations is also low. In the first interval of Figure 7.31 50 percent and more of all sensors observe an event. The overall reset rate is also very low in these intervals. The number of active sensors increases after the last visitor has left. It would be wrong to assume that an increased number of assigned sensors and a high amount of

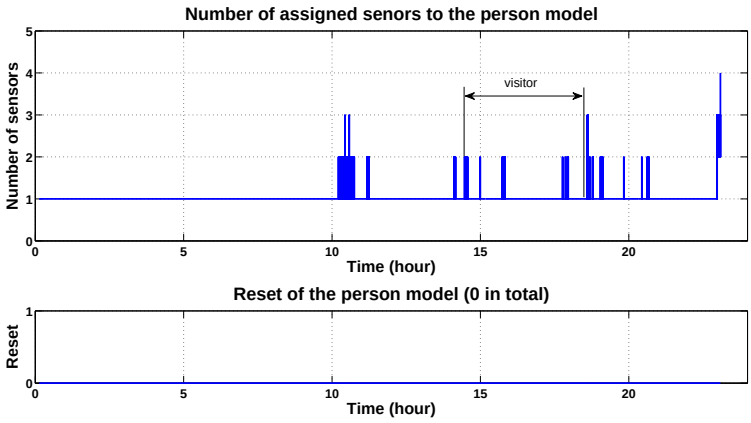


Figure 7.30: Temporal behavior in presence of a visitor (TP3)

Figure 7.31: Temporal behavior in presence of a visitor (TP4)

reset are indicative of more persons in the flat. However, any difference to situations with just one person in the room is difficult to observe. The final summary will give a variety of conclusions when one ore more persons are present in the flat:

One person: The reset count is an indication of the quality of the topology. The situation of getting stuck at an area with incorrect activations does not occur frequently. Moreover, these incorrect events are repressed and do not contribute to the reset actions. The topology is the more decisive element of the process which also causes the resets. A representative topology corrects sensor faults and produces a low amount of reset actions. In comparison, a corrupted representation delivers more resets and the person model may also have more sensors included.

More persons: A greater number of persons in the flat causes more frequently resets of the person model compared to a single person. When the method used insufficient topologies fewer reset actions are present at this condition. It also depends on the persons' individual movements. Persons that are always in the same area cannot be distinguished. The general identification of additional persons cannot be done with these parameters.

8 Conclusion and outlook

Finally, the results and the achieved goals in respect to the aims of this work (see Section 1.3) are discussed. A short summary is presented which describes the key facts. The facts are discussed in Sections 8.1.1 and 8.1.2 in more detail. The problems faced during the design- and implementation phase will also be highlighted. Other possibilities to solve the individual problems are given. In the end proposals for further implementations and use of this work are presented.

8.1 Achieved goals

At the raw data level four different algorithms have been evaluated. The most appropriate in this context is the congeneric multi-sensor data fusion algorithm. The output compared to the other algorithms is similar. The advantages over the other algorithms are the reduced configuration work and the better computational performance (see Table 7.1). The varying weights reflect the fault tolerant behavior. All of the presented algorithms reduce uncertainties when more than one sensor is used.

The concepts to increase the usability are considered in the second part of the thesis. Sensor types of a node are identified by their data values. High classifications rate are achieved by the individual nodes. When the sensor classifications are exchanged among the nodes, 100 percent classification rates are achieved on every day within the recording interval.

After identifying all sensors, they could be used for their individual purpose. The movement sensors were used to gain a flat's topology. The topologies of three different flats have been investigated.

The person model was introduced in order to include the current position of a person. The combination of the person model and the topology introduced fault tolerance to the process. The correction of the spatial information was utilized by implementing the scenarios of Section 5.5. The improved position information depends on the quality of the topology. When an accurate topology is present, incorrect sensors activations have been excluded from the person model.

8.1.1 Raw data fusion

One major part of this thesis is the fusion of raw data into one representation. The requirements for the fusion process were fault tolerance and increased confidence over the measurements. Due

to the spatial arrangement of the nodes and only slight deviations in the measured properties, the confidence weighed fusion algorithm has been introduced. This algorithm applies to implementations in which sensors are arranged in a replicated way. This ensures that all sensors face the same conditions. Moreover, Equation 5.4 confirms that this algorithm lowers the uncertainties of the individual sensors. The requirement of increased confidence is met but fault tolerance is not supported.

Therefore, three extensions of this algorithm have been presented. These algorithms handle sensor faults in different ways. However, the adaptive algorithm is not given in any other work but could be gained from using a distance measurement. The distance measurement concept has already been used in the voting algorithm. The voting algorithm was implemented differently compared to the algorithm in [DLC05]. The progressing steps to fit the needs of this thesis are highlighted in Section 5.1.2.

The comparison of the algorithms showed that they offer similar fused results. At a normal operation[1] the voting- and adaptive algorithm do not exclude any measurements. Thus, only the confidence weighted fusion algorithm is executed. In presence of erroneous measurements the best performing algorithm has been evaluated.

Some measures had to be found which give detailed information about the performance of the individual algorithms. The congeneric multi-sensor fusion algorithm performed best in terms of number of iterations and number of sensors. The other algorithms had to be executed several times, depending on the number of incorrect measurements (see Table 7.1). The weights in the voting- and adaptive algorithm take different sensor placements and uncertainties into account. However, this method would require knowledge of the exact position of each sensor. It is assumed that no information about the location is present.

Another benefit of the chosen algorithms is that the exact sensor parameters do not have to be known a prior. Therefore, the configuration effort is kept low.

The process is used to fuse data from temperature sensors. When comparing the fused temperature to threshold values, too low or too high temperatures can be determined (see Figure 7.12). These conditions are expressed by features. The authors of [LD07] obtain these features at first and fuse the features afterwards. In their work they used an adapted version of the distributed Neymann-Pearson detection method. The features are associated with probabilities. The algorithms of this thesis are more flexible since the fused data is also part of the raw data level. When the behavior analysis demands features, they can be gained in an additional step.

The congeneric multi-sensor data fusion algorithm is dealing with sensor faults. There is no need that the majority of all sensors are working correctly. The algorithm also works for one correct and one incorrect sensor. In case of low sensor density, the values of just two sensors are fused. Compared to the other algorithms the performance is better for the AAL application. The other algorithms need additional iterations to exclude incorrect measurements. The congeneric multi-sensor data fusion algorithm excludes the incorrect sensor, assigning lower weights compared to correct measurements.

8.1.2 Increasing usability

The second part discussed the methods to lower the configuration work during the installation phase. The intention was to identify all sensor types of a node automatically and use the move-

[1] normal operation - operation without sensor faults

ment sensors to obtain the topology of a flat. The topology was used to discriminate between correct and incorrect activations.

Node classification

Node classification has been done using of the data ranges of the individual types. When an insufficient amount of data had been present, the classification delivered an incorrect type. This was caused by the data, within the interval of another sensor. The situation is often faced when discriminating between accelerometers and light sensors (see Figure 5.4).

The results showed that only two to three sensors could not be recognized on average in the whole flat. This number is caused by rare activations of reed contacts. As discussed earlier on, the problem is that some doors stay in the same condition for a long time.

The classification rate is very high which is confirmed by Figure 7.17. Two sensors are not classified correctly on average. The other flats show 100 percent classification rates.

The problem with regard to rare events is that it would take a considerable time to recognize and classify the associated sensors. Therefore, Sensor Fusion allowed the shortening of this time to just one day. A single day is the shortest observation period. However, the fusion process reduced the highest learning interval of 43 days to one day. It is a huge improvement compared to the situation without fusion.

Schemas such as those illustrated in Figure 7.15 show another benefit of this method. At days with no activity, only the time-triggered sensor readings are transmitted to the central unit. Therefore, the differentiation between intervals in which the person is not in the flat is possible. In the representative figures the observation has been done on single days. The interval can also be restricted to smaller intervals. Moreover, sensor faults also influence on the recognition of sensors. In case of a movement sensor, faults (e.g. solar radiation) benefit the process. Other types such as temperature sensors suffer from incorrect classifications and may be misclassified.

The additional benefit of Sensor Fusion is that all sensors can be classified correctly. The classification process would be more difficult when discriminating between binary sensors only. Therefore, the temporal distributions of sensor activations have to be studied. Patterns of different sensors would look very similar. The whole process would even get more difficult when the number of different types increases.

Topology learning

Learning the topology has been a difficult part of the thesis. The implementation with respect to the goals of the thesis has been done by applying the Boltzmann machine learning approach. The geometric conditions do not have to be prior known. During the implementation and testing phase some problems were faced. The important issues are listed in the following:

Incorrect events: Erroneous sensor readings produce correlations between sensors that might have no relationship to each other. This observation was made in the flat of test person 2. Sensors 145 and 148 showed correlation which was not representing the real topology. These false positive events have mainly been due to solar radiation. The simple solution was to use the information at the times of the day when the sun light could not affect the sensors.

Insufficient data from accelerometers: The task was to gain the topology by using the information of the movement sensors. Faulty sensors corrupt the whole process. Therefore, the idea of adding the accelerometer data has been proposed. It turned out that this approach was not practicable because not enough data (e.g. 4 sensor activations in total at a specific day) was available.

Incomplete diaries: The topology can only be gained when only one person is present in the flat. However, the times when having visitors were present have not always been correct and complete enough. The second part of Section 7.4 illustrated that there is hardly any difference between one person and more persons present in the room. There should be some reliable indicators to do the configuration without the use of diaries.

Complex data structures: Chapter 6.2 has already discussed the issues of pairwise correlations. The compatibility matrix of one sensor pair is mapped into a matrix. Every combination forms another matrix, including these compatibility matrices. This is done at a single day, so every day of observation contains the combined matrices. However, the complexity reduced in every progressing step. The resulting representation is a matrix containing zeros and ones.

After solving these issues, representative topologies have been gained. The topologies of flat 2 and 4 fit the real arrangement. In contrast to this, the topology learning process of flat 3 issued some disturbances. Node number 108 has a lot of links to other nodes that do not reflect the accurate topology.

A similar method, for learning the topology has already been mentioned in the related work of Chapter 3. As a consequence, the presented method would have to be adapted as already explained. Moreover, the results of the topology are applied in order to detect incorrect sensor event.

The person model has been introduced to represent the current position of a person. Without fault tolerance the model contains all sensor activations.

Combination of the methods

The combination of the person model with the topology showed that incorrect activations can be recognized. The sensors producing these incorrect readings are not accepted by the person model in general. However, if an erroneous sensor can be reached from the current position, no differentiation between true positive and false positive events can be made. This situation is solved after a considerable number of incorrect activations. All scenarios have been described in Section 5.5. The fact of getting trapped in the area of false positive activations has also been addressed in this context. The reset actions are caused by the topology and the complexity of the arranged sensors. In fact, the topology illustrates which path can be taken from one node to another. The topology is learnt during a certain time interval. The correlations reflect the most common paths. Rare ways to reach another sensor have not been taken into account. These situations of rare possible transitions contribute to the reset values. The number of links of a node is determined by the time interval for a person to get from one region into another. If too much time is allowed, many links will be created. Too less time will create fewer links or even no links at all. The learnt topologies of this work are trade-off solutions between these two situations.

With the help of representative topologies such as those of flat 4, erroneous sensor measurements have been determined and corrected. Topology 3 has faced the most resets of the flats because it is the least representative topology.

The performance of this algorithm depends on the topology. The most attention has to be paid during this task.

8.2 Further work

It is obvious that the thesis is not able to cover all possibilities of Sensor Fusion in Ambient Assisted Living systems. Therefore, further implementations have to be provided. Further work and fields that might take advantage of this thesis are highlighted.

8.2.1 Learning of feature thresholds

The thresholds for temperature features are set to constant values (see Figure 7.12). These are recommendations for the maximum and minimum values. Further work could include that these limits are learnt for each person. The thresholds will be different for every room. The limits have to be adjusted to the purpose of the room. If the topology is considered, the term area should be used instead of room.

8.2.2 Behavior analysis

The work of this thesis provides more complete information for the behavior analysis. The fault tolerance in the implementations should lead to a better analysis of a person's behavior. Especially the more accurate movement data is a huge improvement. This has to be evaluated in further work. Implementation possibilities can be found in Chapter 3.

8.2.3 Reliability models

The diary records showed that the hardware components had to be maintained and replaced with new components/nodes. Another improvement of such systems would be the automatic detections of faulty hardware. The node classification process is able to identify nodes when they transmit data to the central unit. Indicators of an incorrect node are the unrecognized temperature sensor or the unrecognized light sensor. Fail-silent behavior has not been observed. There should be other ways to determine this condition.

The model in [DLC05] estimates the number of faulty sensors and calculates the reliability. The model is described via a hidden Markov model. The authors discriminate between transient and permanent faults. The models of these two conditions are combined into one model.

Maintenance works could be scheduled after the system has recognized sensor faults. Sensor Fusion deals with fault tolerance but additional sensors offer increased confidence.

8.2.4 Combination of more sensor information

A drawback of the implemented topology learning process is that only the correlations of the nodes including movement sensors are gained. This is caused by the insufficient data from the accelerometers. When having a better data quality of the accelerometers, the data can be used in order to find more correlations among the nodes. This way, links between sensors can be found which do not contain any movement sensors.

If the hardware is available the correlation problem can be solved by determining the signal strength of nearby sensors. This method has already been proposed in the work of [NV05]. The nodes including movement sensors can be used as local fusion centres (see Figure 4.3.3). The local fusion centres have to determine nearby sensor nodes which have not been considered in the topology yet. Due to higher sensor density, nodes can also be associated with more than one fusion centre. The process would imply that the architecture changes from a *centralized* to a *hierarchical* architecture.

The consideration is to follow the concept of a totally flexible architecture in which no knowledge about the environment is needed. The flexible way of fusion could be extended to the information of the temperature sensors and light sensors.

8.2.5 Integration into home automation

The introduction in Chapter 1 presents the ideal case when Ambient Assisted Living systems are completely integrated into home automation. The investigations of the State of the Art implementation showed that manufactures have their own systems and are less integrating them into home automation.

One could argue that the manufacturers assume that home automation systems are usually not common in a household. This could be due to higher initial costs compared to a conventional electrical installation [17]. Manufacturers have to create their own systems that partially provide services of home automation [LG07].

When a home automation system is available, the integration of an Ambient Assisted Living system should be considered. The main advantages can use already existing sensors and their ability to deliver data. Depending on the required sensor inputs the installation work can be minimized. Therefore, the appropriate interfaces have to be delivered. This also addresses the interoperability of home automation systems with Ambient Assisted Living systems.

One way to improve the integration into home automation, especially in the context of this work, could be the use of the presented method of learning the topology. Therefore, the original Boltzmann machine learning algorithm could be used to get additional relationships that alleviate the configuration work. A potential realisation addresses the relationships between the movement sensors and the lighting. If the nodes of the movement sensors also include light sensors, the sequential switching of only one light at a time would deliver their relationship. In a later step, this information can be used to assign the lights which should light up when movement is observed.

Literature

[AK08] Akselrod, D. ; Kirubarajan, T.: Modified value iteration algorithm and Dynamic Element Matching based MDP for Distributed data fusion and sensor management. In: *Information Fusion, 2008 11th International Conference on*, 2008, S. 1 –8

[BGP+10] Baumgarten, M. ; Guldenring, D. ; Poland, M. ; Nugent, C. ; Hallberg, J.: Embedding Self-Awareness into Objects of Daily Life – The Smart Kettle. In: *Intelligent Environments (IE), 2010 Sixth International Conference on*, 2010, S. 34 –39

[BHG+09] Bieber, G. ; Hoffmeyer, A. ; Gutzeit, E. ; Peter, C. ; Urban, B.: Activity monitoring by fusion of optical and mechanical tracking technologies for user behavior analysis. In: *Proceedings of the 2nd International Conference on PErvasive Technologies Related to Assistive Environments*. New York, NY, USA : ACM, 2009 (PETRA '09). – ISBN 978–1–60558–409–6, S. 45:1–45:6

[BI96] Brooks, R.R. ; Iyengar, S.S.: Robust distributed computing and sensing algorithm. In: *Computer* 29 (1996), jun, Nr. 6, S. 53 –60. – ISSN 0018–9162

[Bis95] Bishop, C. M.: *Neural Networks for Pattern Recognition*. New York NY. : Oxford University Press Inc., 1995

[BO00] Bedworth, M. ; O'Brien, J.: The Omnibus model: a new model of data fusion? In: *Aerospace and Electronic Systems Magazine, IEEE* 15 (2000), apr, Nr. 4, S. 30 –36. – ISSN 0885–8985

[BRG96] Bosse, E. ; Roy, J. ; Grenier, D.: Data fusion concepts applied to a suite of dissimilar sensors. In: *Electrical and Computer Engineering, 1996. Canadian Conference on* Bd. 2, 1996, S. 692 –695 vol.2

[Bru07] Bruckner, D.: *Probabilistic Models in Building Automation: Recognizing Scenarios with Statistical Methods*, TU Vienna, Diss., January 2007

[CGV+11] Chen, M. ; Gonzalez, S. ; Vasilakos, A. ; Cao, H. ; Leung, V.: Body Area Networks: A Survey. In: *Mob. Netw. Appl.* 16 (2011), April, S. 171–193. – ISSN 1383–469X

[COM+06] Chen, P. ; Oh, Songhwai ; Manzo, M. ; Sinopoli, B. ; Sharp, C. ; Whitehouse, K. ; Tolle, O. ; Jeong, Jaein ; Dutta, P. ; Hui, J. ; Schaffert, S. ; Kim, Sukun ; Taneja, J. ; Zhu, B. ; Roosta, T. ; Howard, M. ; Culler, D. ; Sastry, S.: Instrumenting wireless sensor networks for real-time surveillance. In: *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006. – ISSN 1050–4729, S. 3128 –3133

[Das97] Dasarathy, B.V.: Sensor fusion potential exploitation-innovative architectures and illustrative applications. In: *Proceedings of the IEEE* 85 (1997), Januar, Nr. 1, S.

24 –38. – ISSN 0018–9219

[DLC05] Desovski, D. ; Liu, Y. ; Cukic, B.: Linear randomized voting algorithm for fault tolerant sensor fusion and the corresponding reliability model. In: *High-Assurance Systems Engineering, 2005. HASE 2005. Ninth IEEE International Symposium on*, 2005. – ISSN 1530–2059, S. 153 – 162

[DNWZ08] Diermaier, J. ; Neyder, K. ; Werner, Panek P. ; Zagler, W.L.: Distributed Accelerometers as a Main Component in Detecting Activities of Daily Living, 2008

[DSK+07] Danu, D. ; Sinha, A. ; Kirubarajan, T. ; Farooq, M.F. ; Peters, D.: Performance Evaluation of Multi-platform Distributed Data Fusion Methods for Multi-target Tracking. In: *Aerospace Conference, 2007 IEEE*, 2007. – ISSN 1095–323X, S. 1 –14

[Elm02] Elmenreich, W.: *Sensor Fuson in Time-Triggered Systems*, TU Vienna, Diss., 10 2002

[EP03] Kap. Distributedd Sensor Fusion Networks In: Elmenreich, W ; Peti, P.: *Intelligent Engineering Systems at the Service of Mankind*. Bd. 1. 2003, S. 335–347

[GWET05] Gorski, J. ; Wilson, L. ; Elhajj, I.H. ; Tan, Jindong: Data fusion and error reduction algorithms for sensor networks. In: *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, S. 610 – 615

[HL97] Hall, D.L. ; Llinas, J.: An introduction to multisensor data fusion. In: *Proceedings of the IEEE* 85 (1997), Januar, Nr. 1, S. 6 –23. – ISSN 0018–9219

[ILC09] Ickowicz, A. ; Le Cadre, J.-P.: Bi-target tracking within a binary sensor network. In: *Information Fusion, 2009. FUSION '09. 12th International Conference on*, 2009, S. 412 –419

[Kha03] Khandpur, R. S.: *Handbook of Biomedical Instrumentation*. 2. McGraw-Hill Education (India) Pvt Ltd, 2003. – 944 S. – ISBN 0070473552, 9780070473553. – ISBN 9780070473553

[LD07] Liu, J. ; Du, Q.: A Multi-Sensor Fusion Method Applied to Stability Diagnosis Based on Distributed Neyman-Pearson Algorithm. In: *Intelligent Information Hiding and Multimedia Signal Processing, 2007. IIHMSP 2007. Third International Conference on* Bd. 2, 2007, S. 651 –654

[LG07] Litz, L. ; Gross, M.: Covering Assisted Living Key Areas based on Home Automation Sensors. In: *Networking, Sensing and Control, 2007 IEEE International Conference on*, 2007, S. 639 –643

[LSC01] Lucey, S. ; Sridharan, S. ; Chandran, V.: Improved speech recognition using adaptive audio-visual fusion via a stochastic secondary classifier. In: *Intelligent Multimedia, Video and Speech Processing, 2001. Proceedings of 2001 International Symposium on*, 2001, S. 551 –554

[LT08] Li, H. ; Tan, J.: Body sensor networks based sensor fusion for cardiovascular biosignal predictions. In: *Proceedings of the 2nd International Workshop on Systems and Networking Support for Health Care and Assisted Living Environments*. New York, NY, USA : ACM, 2008 (HealthNet '08). – ISBN 978–1–60558–199–6, S. 3:1–3:6

[LX10] Lu, G. ; Xue, W.: Adaptive Weighted Fusion Algorithm for Monitoring System of Forest Fire Based on Wireless Sensor Networks. In: *Computer Modeling and Simulation, 2010. ICCMS '10. Second International Conference on* Bd. 4, 2010, S. 414 –417

[MCM+11] Mulvenna, M. ; Carswell, W. ; Mccullagh, P. ; Augusto, J.C. ; Zheng, Huiru ; Jeffers, P. ; Wang, Haiying ; Martin, S.: Visualization of data for

ambient assisted living services. In: *Communications Magazine, IEEE* 49 (2011), Nr. 1, S. 110 –117. – ISSN 0163–6804

[Mit07] Mitchell, H.B.: *Multi-Sensor Data Fusion - An Introduction.* Springer Berlin Heidelberg New York, 2007

[Mit08] Mitterbauer, J.: *Behavior Recognition and Prediction in Building Automation Systems*, TU Vienna, Diplomarbeit, November 2008

[Mov98] Movellan, P.: Robust Sensor Fusion: Analysis and Application. In: *MACHINE LEARNING*, 1998, S. 85–100

[Mur96] Murphy, R.R.: Biological and cognitive foundations of intelligent sensor fusion. In: *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 26 (1996), jan, Nr. 1, S. 42 –51. – ISSN 1083–4427

[NAQB+10] Noury, N. ; Anh Quach, K. ; Berenguer, M. ; Teyssier, H. ; Bouzid, M.-J. ; Goldstein, L. ; Giordani, M.: Ubiquitous but non invasive evaluation of the activity of a person from a unique index built on the electrical activities on the residential power line. In: *Proc. 12th IEEE Int e-Health Networking Applications and Services (Healthcom) Conf*, 2010, S. 1–6

[NLF07] Nakamura, E. F. ; Loureiro, A. A. F. ; Frery, A. C.: Information fusion for wireless sensor networks: Methods, models, and classifications. In: *ACM Comput. Surv.* 39 (2007), September. – ISSN 0360–0300

[NV05] Niu, R. ; Varshney, P. K.: Distributed detection and fusion in a large wireless sensor network of random size. In: *EURASIP J. Wirel. Commun. Netw.* 2005 (2005), September, S. 462–472. – ISSN 1687–1472

[OE07] Ozertem, U. ; Erdogmus, D.: Information Regularized Maximum Likelihood for Binary Motion Sensors. In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on* Bd. 2, 2007. – ISSN 1520–6149, S. II–1021 –II–1024

[PA06] Polychronopoulos, A. ; Amditis, A.: Revisiting JDL model for automotive safety applications: the PF2 functional model. In: *Information Fusion, 2006 9th International Conference on*, 2006, S. 1 –7

[PBU10] Peter, C. ; Bieber, G. ; Urban, B.: Affect- and behaviour-related assistance for families in the home environment. In: *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments.* New York, NY, USA : ACM, 2010 (PETRA '10). – ISBN 978–1–4503–0071–1, S. 47:1–47:5

[PBV+09] Park, K. ; Becker, E. ; Vinjumur, J. K. ; Le, Z. ; Makedon, F.: Human behavioral detection and data cleaning in assisted living environment using wireless sensor networks. In: *Proceedings of the 2nd International Conference on PErvasive Technologies Related to Assistive Environments.* New York, NY, USA : ACM, 2009 (PETRA '09). – ISBN 978–1–60558–409–6, S. 7:1–7:8

[PCH08] Picus, C. ; Cambrini, L. ; Herzner, W.: Boltzmann Machine Topology Learning for Distributed Sensor Networks Using Loopy Belief Propagation Inference. In: *Machine Learning and Applications, 2008. ICMLA '08. Seventh International Conference on*, 2008, S. 344 –349

[Rao10] Raol, J. R.: *Multi-Sensor Data Fusion with MATLAB.* Taylor and Francis Group, 2010

[RHS10] Rahman, A. S. M. M. ; Hossain, M. A. ; Saddik, A. E.: Spatial-geometric approach to physical mobile interaction based on accelerometer and IR sensory data fusion. In: *ACM Trans. Multimedia Comput. Commun. Appl.* 6 (2010), November, S. 28:1–28:23. – ISSN 1551–6857

[RM04] Retscher, G. ; Mok, E.: Sensor Fusion and Integration using an adapted Kalman filter approach for modern navigation systems Department of Applied and Engineering Geodesy, Vienna University of Technology; Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, 2004

[SB01] Kap. Revisions to the JDL Data Fusion Model In: Steinberg, A. ; Bowman, C.: *Handbook of Multisensor Data Fusion*. 1. CRC Press LLC, 2001, S. 45–67

[SBW99] Steinberg, A. N. ; Bowman, C. L. ; White, F. E.: Revisions to the JDL Data Fusion Model. In: *Storage and Retrieval for Image and Video Databases*, 1999

[Sch06] Schörgendorfer, A.: *Extended Confidence-Weighted Averaging in Sensor Fusion*. Kolbegasse 44 Haus 9, 1230 Wien, Technical University of Vienna, Diplomarbeit, 5 2006

[SDFGB08] Sun, H. ; De Florio, V. ; Gui, N. ; Blondia, C.: Towards Building Virtual Community for Ambient Assisted Living. In: *Parallel, Distributed and Network-Based Processing, 2008. PDP 2008. 16th Euromicro Conference on*, 2008. – ISSN 1066–6192, S. 556 –561

[SDFGB09] Sun, H. ; De Florio, V. ; Gui, N. ; Blondia, C.: Promises and Challenges of Ambient Assisted Living Systems. In: *Information Technology: New Generations, 2009. ITNG '09. Sixth International Conference on*, 2009, S. 1201 –1207

[SG06] Schiff, J. ; Goldberg, K.: Automated Intruder Tracking using Particle Filtering and a Network of Binary Motion Sensors. In: *Automation Science and Engineering, 2006. CASE '06. IEEE International Conference on*, 2006, S. 580 –587

[SQSX10] Shu Qing, L. ; Sheng Xiu, Z.: A congeneric multi-sensor data fusion algorithm and its fault-tolerance. In: *Computer Application and System Modeling (ICCASM), 2010 International Conference on* Bd. 1, 2010, S. V1–339 –V1–342

[TJDS09] Teixeira, T. ; Jung, D. ; Dublon, G. ; Savvides, A.: Identifying people in camera networks using wearable accelerometers. In: *Proceedings of the 2nd International Conference on PErvasive Technologies Related to Assistive Environments*. New York, NY, USA : ACM, 2009 (PETRA '09). – ISBN 978–1–60558–409–6, S. 20:1–20:8

[Vod08] Vodjdani, N.: The ambient assisted living joint programme. In: *Electronics System-Integration Technology Conference, 2008. ESTC 2008. 2nd*, 2008, S. 1 –2

[YB09] Yin, G. ; Bruckner, D.: *Project ATTEND - AdapTive scenario recogniTion for Emergency and Need Detection*. 2009. – Poster: Ambient Assisted Living (AAL) Forum, Wien (eingeladen); 29.09.2009 - 01.10.2009.

[YB10a] Yin, G. ; Bruckner, D.: Daily activity learning from motion detector data for Ambient Assisted Living. In: *Human System Interactions (HSI), 2010 3rd Conference on*, 2010, S. 89 –94

[YB10b] Yin, G. ; Bruckner, D.: Split-merge algorithm and Gaussian mixture models for AAL. In: *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, 2010, S. 2314 –2318

[ZC04] Zou, Y. ; Chakrabarty, K.: Sensor deployment and target localization in distributed sensor networks. In: *ACM Trans. Embed. Comput. Syst.* 3 (2004), February, S. 61–91. – ISSN 1539–9087

[ZS09] Zhu, C. ; Sheng, W.: Multi-sensor fusion for human daily activity recognition in robot-assisted living. In: *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*. New York, NY, USA : ACM, 2009 (HRI '09). – ISBN 978–1–60558–404–1, S. 303–304

[ZTG+10] Zhang, L. ; Tamminedi, T. ; Ganguli, A. ; Yosiphon, G. ; Yadegar, J.: Hierarchical multiple sensor fusion using structurally learned Bayesian network. In:

Wireless Health 2010. New York, NY, USA : ACM, 2010 (WH '10). – ISBN 978–1–60558–989–3, S. 174–183

[ZW10] Zhang, C. ; Wang, H.: Decentralized Multi-sensor Data Fusion Algorithm Using Information Filter. In: *Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on* Bd. 1, 2010, S. 890 –893

Internet References

[1] *Ambient Assisted Living.* Sensengasse 1, 1090 Vienna : FFG - Austrian Research Promotion Agency, . – http://rp7.ffg.at/ikt_aal; accessed May 2011

[2] Waibel, Uli: *Feasibility Study sozioökonomischer Parameter für die nationale Implementierung von AAL.* – http://rp7.ffg.at/upload/medialibrary/INNOVENDO_Feasibility_Study_dt_AAL.pdf; accessed May 2011

[3] Floeck, Martin: *Concept and Design of an AAL Home Monitoring System based on a Personal Assistive Unit.* – http://www.eit.uni-kl.de/litz/assisted_living/historie/20071122_ami07.pdf; accessed April 2011

[4] *Interim Evaluation Of the Ambient Assisted Living Joint Programme.* – http://ec.europa.eu/information_society/activities/einclusion/docs/aal/interim_evaluation_report.pdf; accessed May 2011

[5] *Erste Zwischenbewertung des gemeinsamen Programms "Umgebungsunterstütztes Leben" (AAL JP).* Europäische Kommission, . – http://www.kowi.de/textonly/Portaldata/2/Resources/fp7/com-2010-aal-interim-de.pdf; accessed July 2011

[6] *CAPSIL Wiki Copy Appendix II - homepage content.* – http://capsil.org/files/AnnexII-CAPSIL-Wiki-Copy-01-06-2009.pdf; accessed May 2011

[7] *CEIT RALTEC - homepage.* – http://www.ceit.at/ceit-raltec; accessed Oct. 2011

[8] *Ambient Assisted Living Joint Programme - homepage.* – http://www.aal-europe.eu/; accessed Nov. 2011

[9] *Europäisiches Forschungsrahmenprogramm: Aktuelles - Digital Agenda.* Sensengasse 1, 1090 Vienna : FFG - Austrian Research Promotion Agency, . – http://rp7.ffg.at/ikt1_17012011; accessed May 2011

[10] ; FFG - Austrian Research Promotion Agency (Veranst.): *Ambient Assisted Living Joint P AAL JP, THE FIRST CALL FOR PROPOSALS AAL-2008-1.* – http://www.ffg.at/sites/default/files/downloads/call/aal_projekte_1_call.pdf; accessed May 2011

[11] *CARE Project Wiki.* – http://wiki.care-aal.eu/index.php/Main_Page; accessed July 2011

[12] *SOFTCARE - homepage.* – http://deutsch.ceit.at/ceit-raltec/projekte/softcare; accessed Oct. 2011

[13] Rougier, Caroline ; Meunier, Jean: *Fall Detection Using 3D Head Trajectory Extracted From a Single Camera Video Sequence.* – http://computer-vision.org/4security/pdf/montreal-fall_detection.pdf; accessed July 2011

[14] Kluge, Sebastian: *Moderne Navigationssysteme.* 2010. – http://www-m6.ma.tum.de/foswiki/pub/M6/Lehrstuhl/SebastianKluge/Moderne_Navigationssysteme.pdf; lecture notes accessed May 2011

[15] *Sensor Fusion*. Danish GPS Center . – `http://old.gps.aau.dk/downloads/fusion.pdf`; accessed May 2011

[16] *Behaglichkeit*. – `http://www.staubi.at/media/sunswissbehaglichkeit.pdf`; accessed Aug. 2011

[17] *Home Automation Costs - homepage*. – `http://www.homesecurityinformation.com/home-automation-costs.htm`; accessed Oct. 2011