# Vorlesung: Collaborative transmission in wireless sensor networks

Sommersemester 2010

Version: April 9, 2010 ($v0.0.2 + \varepsilon$)

Veranstalter: Stephan Sigg

Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Verteilte und Ubiquitäre Systeme

D-38106 Braunschweig

# Acknowledgements

I would like to thank the students of my lecture 'Collaborative transmission in wireless sensor networks' at the Technische Universität Braunschweig in the winter term 2009/2010 for their patience, thoughtful questions and constructive feedback and discussions on the topic. In particular, Timo Schulz and Sascha Lity have spotted many spelling errors in prior versions of this document.

Some findings presented in this document are the result of student thesis at the Technische Universität Braunschweig in 2009. Section 8.3.3 was predominantly influenced by the work of Rayan Merched El Masri. The results in section 8.4.2 have been derived by Julian Ristau.

# Contents

# Abbreviations and Notation

The following notations are utilised throughout this document. It has been attempted to keep the standard notation from the literature whenever possible. However, since diverse scientific areas are covered, the notation had to be adapted in order to provide an unambiguous notation. The page number given in the table refers to the first occurrence of the mentioned construct.

| Notation | Explanation | Page |
|---|---|---|
| $A$ | Region where nodes of a sensor network are placed | 42 |
| $\alpha$ | Path-loss exponent | 52 |
| $B$ | Bandwidth | 55 |
| CDMA | Code division multiple access | 14 |
| CML | Context Modelling Language | 24 |
| $c$ | Speed of light $(3 \cdot 10^8 \frac{m}{s})$ | 49 |
| $d$ | Distance | 51 |
| $E[x]$ | The expectation of a random variable $x$ | 68 |
| $\eta$ | Density of a sensor network | 42 |
| $f$ | Frequency | 49 |
| $\mathcal{F}$ | Fitness function | 84 |
| $G_{RX}$ | Gain of the receive antenna | 51 |
| $G_{TX}$ | Gain of the transmit antenna | 51 |
| GPS | Global Positioning System | 17 |
| GSM | Global System for Mobile Communications | 17 |
| $\gamma$ | Phase offset of a transmit signal | 49 |
| $\Im(s)$ | Imaginary part of a complex signal $s$ | |
| $IAC$ | inquiry access codes | 57 |
| ID | Identification | 27 |
| IR | infra-red | 25 |
| $ISM$ | Industrial, Scientific, Medical band | 57 |
| i.i.d. | Identically and independently distributed | 14 |
| $J$ | Joule | 55 |

| Notation | Explanation | Page |
|---|---|---|
| $K$ | Kalvin | 55 |
| $\kappa$ | Boltzmann constant | 55 |
| $\lambda$ | Wavelength of a transmit signal | 49 |
| MIMO | Multiple input multiple output | 13 |
| MISO | Multiple input single output | 60 |
| MIT | Massachusetts Institute of Technology | 25 |
| $\mu$ | Population size of an evolutionary algorithm | |
| $\mu(R)$ | Density of a sensor network with transmission range $R$ | |
| $N$ | Network size | 42 |
| $\nu$ | Offspring population size of an evolutionary algorithm. Note: In the literature, typically $\lambda$ denotes the offspring population size | 74 |
| $P_N$ | Thermal noise power | 55 |
| $P_{RX}$ | Received signal power | 51 |
| $P_{TX}$ | Transmission power | 51 |
| $P(x)$ | Probability of an event $x$ | 66 |
| $P(\chi_1\|\chi_2)$ | Conditional probability of 2 events $\chi_1$, $\chi_2$ with $P(\chi_2) > 0$ | 68 |
| $\mathcal{P}$ | An optimisation problem | 84 |
| $\Pi$ | Sample space | 66 |
| $\Re(s)$ | Real part of a complex signal $s$ | 50 |
| $R_k(t)$ | The reliability or fault tolerance of a sensor node | 41 |
| $R$ | Transmission range of a sensor network | 42 |
| RF | Radio frequency | 14 |
| RMSE | Root of the Mean Squared Error | 111 |
| $RSS$ | Received Signal Strength | 51 |
| $S$ | A search space | 73 |
| SIMO | Single input multiple output | 60 |
| SINR | Signal to interference and noise ratio | 56 |
| SISO | Single input single output | 61 |
| SNR | Signal to noise ratio | 15 |
| $s^*$ | Complex conjugate of $s$ | |
| $T$ | Temperature in Kalvin | 55 |
| UbiComp | Ubiquitous Computing | 18 |
| UMTS | Universal Mobile Telecommunications System | 17 |
| $\upsilon$ | Failure rate | 42 |
| $var[\chi]$ | The variance of a random variable $\chi$: $E[(\chi - E[\chi])^2]$ | 69 |
| $\overrightarrow{v}$ | A vector $v = (v_1, \ldots, v_k)$ | |

| Notation | Explanation | Page |
|---|---|---|
| WLAN | Wireless Local Area Network | 17 |
| WSN | Wireless sensor network | 14 |
| $x$ | A sample point for a random experiment | 64 |

# 1 Motivation

> *In the long history of humankind (and animal kind, too) those who learned to collaborate and improvise most effectively have prevailed*
>
> (C. Darwin)

In recent years, sensor nodes of extreme tiny size are envisioned [1, 2, 3]. In [4], for example, applications for square-millimetre sized nodes that seamlessly integrate into an environment are detailed. At these small form-factors transmission power of wireless nodes is restricted to several microwatts. Communication between a single node and a remote receiver is then only feasible at short distances. It is possible, however, to increase the maximum transmission range by cooperatively transmitting information from distinct nodes of a network [5, 6]. The basic idea is to superimpose identical RF carrier signal components from various transmitters that function as a distributed beamformer. When the relative phase offset of these carrier signal components at a remote receiver is small, the signal strength of the received sum signal is improved. Cooperation can improve the capacity and robustness of a network of transmitters [7, 8] and decreases the average energy consumption per node [9, 10, 11].

Related research branches are cooperative transmission [12], collaborative transmission [13, 14], distributed adaptive beamforming [15, 16, 17, 18], collaborative beamforming [19] or cooperative/virtual MIMO for wireless sensor networks [20, 21, 22, 23]. One approach is to utilise neighbouring nodes as relays [24, 25, 26] as proposed by Cover and El Gamal in [27]. Cooperative transmission is then achieved by Multi-hop [28, 29, 30] or data flooding [31, 32, 33, 34] approaches. The general idea of multi-hop relaying based on the physical channel is to retransmit received messages by a relay node so that the destination will receive not only the message from the source destination but also from the relay. In data flooding approaches, a node will retransmit a received message at its reception. It has been shown that the approach outperforms non-cooperative multi-hop schemes significantly. It was derived that the average energy consumption of nodes is decreased [9, 10] and the transmission time is reduced compared to traditional transmission protocols in wireless sensor networks [35].

In these approaches, nodes are not tightly synchronised and transmission may be asynchronous. This, however, is achieved by virtual MIMO techniques. In virtual MIMO for wireless sensor networks, single antenna nodes are cooperating to establish a multiple an-

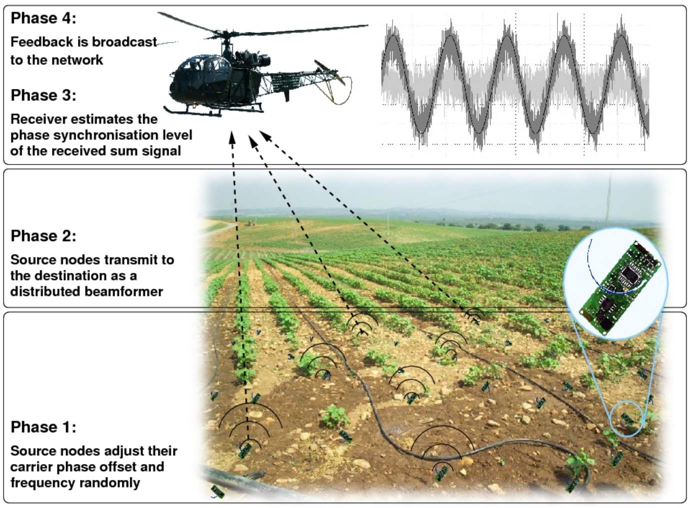


Figure 1.1: Schematic illustration of feedback based distributed adaptive beamforming in wireless sensor networks

tenna wireless sensor network [21, 20, 22]. Virtual MIMO has capabilities to adjust to different frequencies and is highly energy efficient [23, 11]. However, the implementation of MIMO capabilities in WSNs requires accurate time synchronisation, complex transceiver circuits and signal processing that might surcharge the power consumption and processing capabilities of simple sensor nodes.

Other solutions proposed are open-loop synchronisation methods as round-trip synchronisation based [36, 37, 38]. In this scheme, the destination sends beacons in opposed directions along a multi-hop circle in which each of the nodes appends its part of the overall message to the beacons. Beamforming is achieved when the processing time along the multi-hop chain is identical in both directions. This approach, however, does not scale well with the size of a network.

Closed loop feedback approaches include full-feedback techniques, in which carrier synchronisation is achieved in a master-slave manner. The phase-offset between the destination and a source node is corrected by the receiver node. Diversity between RF-transmit signal components is achieved over CDMA channels [39]. This approach is applicable only to small network sizes and requires sophisticated processing capabilities at the source nodes.

A more simple and less resource demanding implementation is the one-bit feedback based closed-loop synchronisation considered in [39, 40]. The authors describe an iterative process in which $n$ source nodes $i \in [1, \dots, n]$ randomly adapt the phases $\gamma_i$ of their carrier signal $\Re\left(m(t)e^{j(2\pi(f_c+f_i)t+\gamma_i)}\right)$. Here, $f_i$ denotes the frequency offset of node $i$ to a common carrier frequency $f_c$. Initially, i.i.d. phase offsets $\gamma_i$ of carrier signals are assumed. When a receiver requests a transmission from the network, carrier phases are synchronised in an iterative process (cf. figure 7.10).

1. Each source node $i$ adjusts its carrier phase offset $\gamma_i$ and frequency offset $f_i$ randomly.

2. The source nodes transmit to the destination simultaneously as a distributed beamformer.
3. The receiver estimates the level of phase synchronisation of the received sum signal (e.g. by the SNR).
4. This value is broadcast as a feedback to the network. Nodes interpret this feedback and adapt their phase adjustments accordingly.

These four steps are iterated repeatedly until a stop criteria is met (e.g. maximum iteration count or sufficient synchronisation). The process has been studied by various authors [41, 42, 43, 13] where the approaches proposed differ in the implementation of the first and the fourth step specified above. The authors of [43] show that it is possible to reduce the number of transmitting nodes in a random process and still achieve synchronisation among all nodes.

In [41, 42, 43] a process is described in which each node alters its carrier phase offset $\gamma_i$ according to a normal distribution with small variance in step one. In [13] a uniform distribution is utilised but the probability for one node to mutate is low. Only in [42] not only the phase but also frequency is adapted.

This lecture is focused on cooperative transmission schemes in wireless sensor networks. Distinct nodes in a network of nodes cooperate in their transmission of data. As detailed above, this cooperation may differ in its exact implementation for various approaches. Some approaches require inter-node communication while others don't. For some approaches the aim is to reduce the failure probability through multiple transmissions while others aim to improve the signal strength. Figure 1.2 depicts for the generic scenario introduced above the organisation of the lecture.

First, the theoretical background required to understand cooperative transmission scenarios is provided in chapters 2 through 4. In Chapters 5 and 6, concepts required for the application and analysis of the algorithms for cooperative transmission in wireless sensor networks in chapters 7 and 8 are introduced. The focus of the algorithms presented is on algorithms for distributed adaptive transmit beamforming.

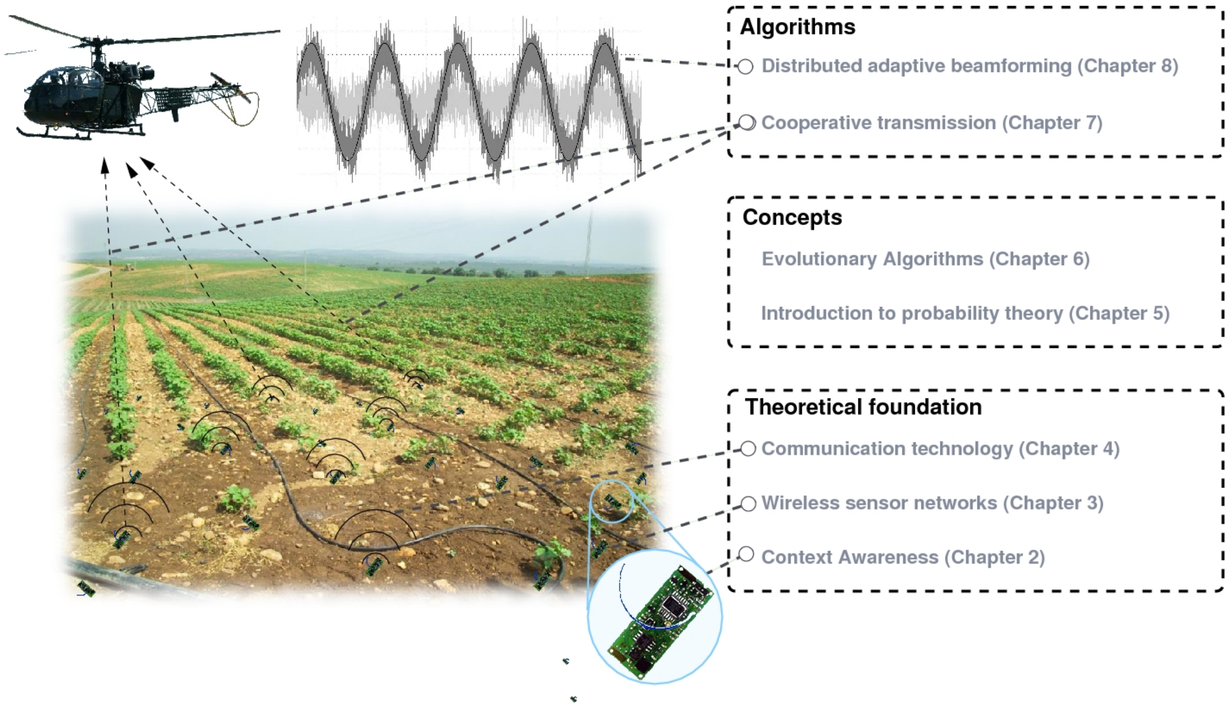


Figure 1.2: Possible scenario for distributed adaptive transmit beamforming

# 2 Context-awareness

> *Increasingly, the bottleneck in computing is not its disk capacity, processor speed or communication bandwidth, but rather the limited resource of human attention*
>
> (A. Garlan, Toward Distraction-Free Pervasive Computing [44])

The vision of context-awareness is that applications become sensitive to environmental stimuli and adapt their behaviour to the current situation. This vision was far ahead of the technology of the time when it was first studied in research laboratories and the details necessary to implement it were seldom provided. With improved technology we have seen prototype applications of individual ideas from the Context-aware vision become implemented. The first of these are probably the Xerox PARCTAB [45] and the MediaCup [46].

In recent years, but to a limited degree, we have already seen context-aware features in consumer products. Mobile devices that adjust their screen brightness to the environmental light, devices that automatically rotate the screen when the device is turned, watches that automatically adjust to local time and messages that alert users when their screen work time exceeds a certain limit, are just some examples.

While these applications are quite limited and isolated, we see more advanced and better integrated context-aware features in multifarious new products. The most versatile and widely used device type for context-aware applications are recent mobile phones. The capabilities of these devices quickly increase as new interfaces to the environment are constantly added. Apart from technologies as basic as microphones, speakers and GSM, we now expect also infrared, bluetooth and a camera in mobile devices. New air interfaces as WLAN or UMTS are added, as well as light sensors, accelerometers, touch screens and to an increasing degree GPS receivers. Many of these technologies remain unused most of the time. This multitude of sensors, however, provides a rich environment in which context-aware applications can be taken to the next evolutionary stage. Context-awareness, nowadays, still holds great potential before the development comes anywhere close to the vision of a ubiquitous world that is saturated with context-aware devices.

In recent years, applications and devices have undergone serious changes that move them away from static, reactive entities towards a more environment responsive design. We see applications act in an increasingly adaptive and situation-dependent way. Applications are able to infer the needs and requirements in a given situation. It is commonly agreed that the general setting a user is in also influences her needs at that point in time. Lucy Suchman [47] states that every course of action is highly dependent upon its social circumstances regarding interactions between actors and the environment. To become able to react to the general setting an application is executed in, the design paradigm for applications is shifting from an application-centric approach to an environment-centric approach. Applications become integrated into the environment and react to environmental stimuli. In order to improve the application and device behaviour in this direction, further and in most cases novel sources of information are investigated.

The input provided to an application or device is no longer restricted to explicit instructions on a common user interface. Instead, the interface utilised for the acquisition of input information is extended and coupled by an interface to the environment. The behaviour of applications evolves from a mere passive, input dependent way to an active, environment and situation guided operation.

Information about the environment and situation is extracted and interpreted to trigger situation dependent actions that shall, for example, provide the user with a richer experience that is adapted to her personal needs. Due to this additional information, the required explicit interaction with an application can be minimised or at least reduced. The computing experience hereby becomes increasingly unobtrusive and ubiquitous.

In general, this computing paradigm is referred to as context-awareness or context computing but is described by various further titles. People have been quite creative in finding descriptive names for scenarios similar to the one described above. A (most certainly not exhaustive) set of terms associated with ideas related to context computing is depicted in figure 2.1. A similar list can also be found in [48]

While these catchwords have partly redundant but not identical meanings, a common vision of future computing is captured by all these descriptions. Probably the first study on context-aware computing was the Olivetti Active Badge [49]. Following this pioneering work, numerous further concepts and ideas have been discussed by various research groups.

Recently, the focus for context awareness has shifted from a single device sensing its environment to an environment capable of sensing and possibly interpreting and reacting on the sensed information. Sensor nodes i.e. tiny computing devices with communication and sensing capabilities become integrated in the environment.

## 2.1 Context-aware computing

The vision of a world where computing devices seamlessly integrate into the real world was first introduced by Mark Weiser in 1988. He illustrates and describes his vision of future computing in [50]. Computing in his vision is no longer restricted to a single machine but may move off one machine and onto another one at execution time. Ubiquitous computing also incorporates an awareness of the environment the computer is situated in. Furthermore, following the vision of ubiquitous computing, computing becomes invisible and omnipresent simultaneously. Smallest scale computing devices that enrich the envi-

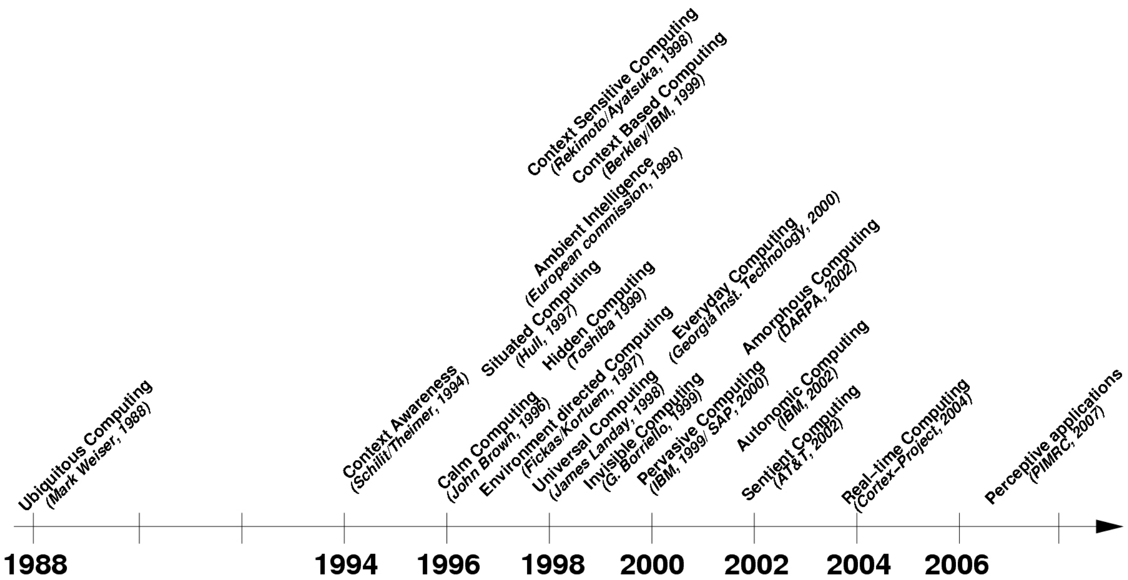


Figure 2.1: Concepts related to Ubiquitous computing

ronment communicate with each other and assist a user unnoticed. Weiser argues that a computer might adapt its behaviour in a significant way if it knows where it is located. As Weiser states, this reaction to the environment does not require artificial intelligence.

Weiser observes the paradox that computing devices are becoming cheaper, smaller and more powerful at the same time. Tiny computing devices become cheap enough to be bought in raw amounts and small enough to be integrated in virtually every real world object.

Weiser envisions that these devices, equipped with sensing technology and communication interfaces are able to communicate with each other and to acquire and spread information on devices, persons and objects in their proximity. This information can then be utilised to enhance the computing experience of a user.

The first experiments with computers aware of their environment have been conducted in the early 1990's. The active badge location system by Olivetti Research [49] and the Xerox PARCTAB location system by Xerox laboratories [45] demonstrated how small mobile devices operate together.

Although the sources of information utilised in these experiments were restricted to location sensors, the basic new concept and possibility inspired numerous people to focus their research on this field.

### 2.1.1 Definitions of context

Definitions of context are numerous and diverse even when the focus is restricted to computer sciences. In his comprehensive discussion "What we talk about when we talk about context"[51] Paul Dourish attempts to exhaustively discuss several aspects of context and also reviews various definitions of context.

The concept of context in conjunction with context-aware computing was first formulated by Schilit and Theimer in 1994 [52]. Following their definition, a software that "adapts according to its location of use, the collection of nearby people and objects as well as changes to those objects over time" is considered to be context-aware. Later on, Schilit refined this definition by defining context categories in [53]. These categories are 'user context', 'physical context' and 'computing context'. As further categories, Brown added information about the time [54], while Pascoe also considered the blood pressure of users [55]. Dey took the latter proposal to a broader scope by considering emotions and the focus of attention [56].

At about the same time, Albrecht Schmidt, Michael Beigl and Hans W. Gellersen recognised that most so-called context-aware applications are in fact location-aware [57]. Hence, they are considering only location as an aspect of the context. The assertion of the authors is that applications implemented on mobile devices might significantly benefit from a wider understanding of context. Furthermore, they introduce a working model for context and discuss mechanisms to acquire other aspects of context beside location.

In their working model for context, they propose that a context describes a situation and the environment a device or user is located in. They state that a context shall have a set of relevant aspects to which they refer as features.

These features are ordered hierarchically. At the top level a distinction between human factors and physical environment is made. Further, finer grained sub-divisions of these top-level categories are also proposed. Finally, an overview of available sensor types and contexts obtained from these sensors is given.

As a prerequisite to a definition of context-awareness, Anind K. Dey formulated a definition of context, that is most commonly used today [58].

**Definition 2.1.1 : User context**

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

This definition, while useful, is quite abstract and gives no hint on the actual representation of context in a computing system. For this reason, several authors criticise it. As Jani Mäntyjärvi stated in [59], this context definition does not result in a more exact definition of context since the abstraction is shifted from context to information.

Karen Henricksen follows the same line of argumentation by remarking that the definition remains too imprecise, since a clear separation of the concepts of context, context modelling and context information is not provided. Henricksen refines the definition of context given by Dey as the set of circumstances surrounding a task that are potentially relevant for its completion [60]. Furthermore, in the model of Henricksen, a context model identifies a subset of the context that is realistically attainable from sensors, applications and users. Following her discussion, context information describes a set of data that was gathered from sensors and users and that conforms to a context model.

However, the discussion about a most suitable definition is not settled yet. In 2000, Lieberman and Selker defined context to be any input other than the explicit input and output [61]. Other projects refine the definition of context to their individual needs. In [62] for example, the definition of Dey is refined by adding the concept of a sentient object.

### 2.1.2 Context-awareness

Intuitively, applications that utilise context data are context-aware. However, similar to the lively discussion on a definition of context, several definitions for context-awareness have been given in the literature. This section briefly reviews this ongoing discussion.

In [52] Schilit and Theimer formulated a first definition of context-awareness. Following this definition, "Applications are context-aware when they adapt themselves to context".

In 1998 Pascoe argues that context-aware computing is the ability of devices to detect, sense, interpret and respond to changes in the user's environment and computing devices themselves [63]. The authors of [64] define context-awareness as the automation of a software system based on knowledge of the user's context. Several other similar definitions treat it as applications' ability to adapt or change their operation dynamically according to the state of the application and the user [52, 54, 65].

Later, Dey argued that the existing definitions did not fit to various applications developed at that time that were intended to be context-aware and consequently stated a more general definition of context-aware systems in [58].

**Definition 2.1.2 : Context-awareness**

> *A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.*

This discussion is not closed yet as several research groups refine the definition so that it best suits their needs (cf. [62]).

### 2.1.3 Context processing

Context is an abstract concept to describe a major input of ubiquitous computing applications. However, it is not possible to build applications with this theoretical construct. To do this, we have to define how context can be obtained from the available information sources, in which way context is represented in applications and how context can be further processed. This section discusses popular approaches to these questions.

Various authors propose to pre-process sensor output in order to prepare the data for further computation. Anind K. Dey argues that one of the main reasons why context is not used in applications is because no common way to acquire and handle context is specified [58]. He proposes to separate the context acquisition from the context utilisation process. Dey distinguishes between two basic forms of context. Raw or low-level context data that is directly acquired by sensors and richer or higher-level forms of information. A similar distinction is also made by Guanling Chen [66]. However, no concrete specification of these notions is given.

Albrecht Schmidt on the other hand argues that it is simpler to implement context-aware systems using contexts on entity level [67]. With the notion 'entity level', Schmidt refers to context data that is not further processed or aggregated after it has been obtained from context sources. Furthermore, intrinsic properties of sensors are utilised in the context modelling process. Schmidt refers to this approach as the concept of bottom-up context-awareness. His main research focus is related to context acquisition from a variety of simple sensors. He defines simple sensors as low-end, low-price computing and communication technology.

These ideas are utilised by Johan Himberg. Himberg studies data mining and visualisation for context-awareness and personalisation [68]. He especially focuses on sensor data captured by on-board sensors of mobile phones. He investigates how to infer context from features derived from the sensor signals. Johan Himberg especially utilises simple statistical methods in order to reach his aim.

An approach focused on the whole process of context inference is proposed by Jani Mäntyjärvi. Mäntyjärvi considers the problem, how low-level contexts can be obtained from raw sensor data [59]. This problem is basically related to the extraction of features from information sources. For each context a set of features is relevant that determines the context. After the feature inference process, Mäntyjärvi composes the sampled features to obtain a more expressive description of a context. This operation is considered as the processing of low-level contexts to obtain high-level contexts.

Mäntyjärvi presents a procedure for sensor-based context recognition. This approach is referred to as bottom-up approach, in contrast to a top-down approach that starts from the high-level context as it had been proposed by Dey in [58]. Included in this procedure is also a method to extract information on contexts and to convert it into a context representation. Following his definition, raw sensor data is data like $24°C$, or 70% humidity. Low-level contexts are defined as pre-processed raw sensor data where the pre-processing may be constituted, for example, from noise removal, data calibration and reforming of data distributions. Generally, low-level contexts are conditions like 'warm' or 'normal humidity'. Higher level contexts are then created by an additional processing of low-level contexts that results in an action like 'having lunch'.

Main assumptions prior to his work are that sensors attached to computing devices have to be carefully chosen in order to be useful and that context actually can be recognised by sensor data.

The term context atom was introduced in [69] and has been used by Jani Mäntyjärvi, Johan Himberg and Pertti Huuskonen to describe basic context dimensions which are derived from low-level sensor data by pre-processing [70].

### 2.1.4 Frameworks and architectures for context-awareness

In order to facilitate the development of context-aware applications, several authors have proposed frameworks and architectures for this task.

In his PhD thesis in 1994 [71], Schilit concludes that traditional software approaches are not well-suited to build distributed mobile systems. The main reason for this dilemma is that applications are seldom designed to adapt their behaviour to the ever-changing mobile environment of a user in which they are executed. By designing an architecture that communicates context changes to the application, Schilit proposes a solution to this problem.

Additionally, Schilit identifies the problem that the user context may not be shared by distinct applications, although they are actually executed in the same user context. Schilit proposes the use of a user agent that administers the user context in order to provide a persistent dynamic context for all applications of the user.

Furthermore, he presents a system structure for use with context-aware systems. He recommends a distribution of system functions and designs protocols for communication between the entities.

These thoughts are further developed in the context toolkit that was introduced in 2000 [58]. It was proposed and developed by Anind K. Dey at the Georgia Institute of Technology. The context toolkit constitutes a conceptual framework that was designed to support the development of context-aware applications. It is widely accepted as a major reference for context-aware computing. An important contribution of this framework is that it distinguishes between context sensing and context computing. Context sensing describes the process of acquiring information on contexts from sensors while context computing refers to the utilisation of acquired contexts. Basic components in this architecture are context widgets (encapsulated sensors), aggregators and interpreters. However, the Context Toolkit is not generally applicable for arbitrary context-aware applications since it exclusively features discrete contexts and does not consider unreliable or unavailable sensor information [72].

Later on, Albrecht Schmidt presented a "working model for context-aware mobile computing" which is basically an extensible tree structure [67]. The proposed hierarchy of features starts with distinguishing human factors and the physical environment and expands from there. One of the major contributions of his PhD thesis is a framework supporting design, simulation, implementation and maintenance of context acquisition systems in a distributed ubiquitous computing environment.

In 2003, Karen Henricksen introduced a novel characterisation of context data in ubiquitous computing environments [60]. Her introductory study of the ubiquitous computing environment especially focuses on challenges in providing computing applications in ubiquitous computing environments. These issues can be summarised as the autonomy of computing applications, dynamic computing environments, dynamic user requirements, scalability and resource limitations. Henricksen concludes that this set of challenges necessitates a new application design approach. Henricksen proposes a conceptual framework and a corresponding software architecture for context-aware application development.

This framework consists of programming models to be used for context-aware systems. Furthermore, Henricksen proposes the use of the Context Modelling Language (CML), a graphical notation of context that supports the specification of application requirements by the application designer.

In 2004 the Solar framework was presented by Chen [66]. It provides means to derive higher-level context from lower level sensor data.

The framework basically represents a network of nodes that interact with each other. It is scalable, supports mobility of nodes and is self managed.

Solar is designed as a service-oriented middleware in order to support the distribution of its components. The middleware supports sensors, as well as applications. Components and functions can be shared between applications. The data flow between sensors and applications may be composed as a multi-layered acyclic directed graph both at design time or at runtime.

Together with Solar, Chen provides a graph-based programming model, that can be utilised for the design of context-aware architectures.

### 2.1.5 Applications utilising context

Several applications that utilise context have been developed in recent years. In this section we introduce a set of applications that illustrate the uses and application fields of context-aware computing applications. The number of context-aware applications has reached an immense quantity. It is beyond the scope of this document to present an exhaustive overview of these applications. The examples presented are chosen in order to illustrate the broad spectrum of approaches and to show the possibilities for context-aware applications.

With the MediaCup [46], Hans W. Gellersen, Michael Beigl and Holger Krall have presented a context-aware device that demonstrates one part of Mark Weiser's vision of ubiquitous computing. The MediaCup is a coffee cup that is enriched with sensing, processing and communication capabilities. The cup was developed to demonstrate how ordinary, everyday objects can be integrated into a ubiquitous computing environment. The context data obtained by the cup is related to the location of the cup, the temperature and some movement characteristics. This information is obtained by a temperature sensor and an acceleration sensor. Context information can be broadcast with the help of an infra-red (IR) diode. The MediaCup has been utilised in research projects in order to provide a sense of a remote presence and in order to log user activity.

Another application proposed by Gellersen et al. is context acquisition based on load sensing [73]. With the help of pressure sensors in the floor of a room, the presence and location of objects and individuals can be tracked. Furthermore, it is shown that it is possible to distinguish between objects and that even movement of objects can be traced. The authors consider the use of load sensing in everyday environments as an approach to acquisition of contextual information in ubiquitous computing systems. It is demonstrated that load sensing is a practical source of contexts. It exemplifies how the position of objects and interaction events on a given surface can be sensed.

Various implemented context-aware applications have been developed by the Context-Aware Computing Group at the MIT[1]. An illustrative example is the 'Augmented Reality Kitchen' that monitors the state of objects in a kitchen in order to help the kitchen-worker to keep track of all simultaneous events. The kitchen displays the location of tools and the state of cooking processes. In the related project 'KitchenSense', a sensor-rich networked kitchen is considered that attempts to interpret peoples' intentions and reacts accordingly.

Additionally, the SenseBoard has been proposed in [74]. The SenseBoard approach is to combine the benefits of the digital world with those of the real world. The SenseBoard is a hardware board with a schedule projected onto it. Discrete information pieces that are stored in a computer can be manipulated by arranging small items on the board. These items are entries of the schedule. The naming of each item is computer-controlled and projected onto the item. Like in a digital schedule, items can be easily arranged, grouped together or expanded. Operations and the status of the schedule are projected to the physical schedule on the board. Like with real-world objects, people can manually arrange the items on the hardware board. This makes the operation more intuitive and enables the participation of larger groups in the process of finding an optimal schedule for a given task. Detailed information on each item can be made available and a schedule can be digitally exported, stored or loaded and also printed.

[1] http://context.media.mit.edu/press/index.php/projects/

## 2.2 Concepts and definitions

As mentioned in section 2.1, the concepts and ideas related to context-awareness that have not yet been commonly adopted among researchers even include the notion of context and context awareness itself. Since context-awareness is a comparably young research field, we find concepts and notions for which a variety of only partially redundant definitions have been given. On the other hand, several supplementing concepts are only vaguely described as, for example, the notion of high-level contexts, low-level contexts and raw data. In order to provide a stringent view on our research topics, we have to agree on non-ambiguous definitions for the concepts we utilise.

In this section we discuss those notions we adopt from recent work and further find comprehensive definitions for insufficiently defined concepts where necessary.

### 2.2.1 Ubiquitous computing

In our view of ubiquitous computing we agree on the vision introduced by Mark Weiser in [50]. It is to assume a world in which computation has both infiltrated everyday life and vanished from people's perception. Some people believe that both developments are not only possible but predefined, since computing devices continuously decrease in size and power consumption while increasing in computing power at the same time. In the vision of ubiquitous computing, everyday objects are equipped with computing power and communication interfaces in order to compute and spread information. In our study we assume that computing is done in a ubiquitous environment, where multiple applications on stationary and mobile devices interact with one another.

Several authors have observed challenges of ubiquitous computing environments. The authors of [60], for example, state increased autonomy, a dynamic computing environment, dynamic user requirements, scalability issues and resource limitations as most serious issues in UbiComp environments. Depending on the application type, further issues may be named.

### 2.2.2 Sensors, context sources and features

In context-aware computing domains, the input data for applications is captured by sensors. Basically, a sensor is a piece of hardware or software that provides information on the environment. Humans or animals are not considered sensors but might trigger and influence sensor outputs. We distinguish between hardware sensors and software sensors. Hardware sensors are physical entities that react to stimuli from the physical environment and provide a software interface to publish notification describing these stimuli. Hardware sensors might, for example, measure the temperature, the light intensity or the humidity. Further hardware sensors are, for instance, a fingerprint reader or also a computer keyboard or a mouse that monitor user input.

Software sensors are applications that react to software generated stimuli and that output a software generated notification describing these stimuli. Example software sensors are a calendar, an address book or an application a user is interacting with.

A sensor might provide various distinct aspects of a given context. Consider, for example, an audio sensor that provides the loudness as well as the number of zero crossings. These distinct aspects of context are often referred to as context features [57, 67]. We are especially interested in the entity that provides information about a context feature.

We refer to this entity as a context source and consider context sources as atomic information sources for context-aware architectures. Context sources are not synonymous to sensors that produce context data. One sensor might incorporate several context sources. A context source basically produces output values that are related to one specific feature of a sensor.

### 2.2.3 Context and context types

As we have discussed in section 2.1.1 various definitions of context have been given in the literature that are only partly redundant. We adopt the definition given by Anind K. Dey in [58] since it is most general and can be applied to all application areas relevant to our research. However, Dey explicitly intertwines context with the interaction of applications and humans or, as he states it, with users. We have a slightly wider understanding of context that is not restricted to the user-application interaction but that covers contexts of arbitrary entities.

**Definition 2.2.3 : Context**

> *Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object.*

Other definitions of context are too restricted to special cases to be applied in our general, computation-centric, consideration. Considering the revised definition given by Karen Henricksen, after which context is the set of circumstances relevant for the completion of a task [60], we disagree.

This revised definition differs from our understanding of context. First of all, we do not agree with the restriction of context to the set of circumstances that are of potential relevance for the completion of a task. The context driving, for example, could be partly sensed through the presence of the bluetooth ID of the car radio. However, the car radio is of no relevance considering the completion of the context driving.

In addition to the general understanding of the concept of context, a more concrete frame is required in order to be able to actually apply computations on context. We introduce the notion of a context element that utilises the definition of Dey and enhances the description to suit our needs in the processing of contexts.

**Definition 2.2.4 : Context element**

> *Let $i \in \mathbb{N}$ and $t_i$ describe any interval in time. A context element $c_i$ is a non-empty set of values that describe a context at one interval $t_i$ in time.*

An example for a context element that is constituted from the temperature, the light intensity and an IP address is then $c = \{24°C, 20000lx, 141.51.114.33\}$. Observe that this definition refers to an interval in time rather than to a point in time. This accounts for the fact that the information describing a context is obtained by measurements of the real world that typically require a time-span rather than a time instant in which the measurement is performed. However, the shorter the time span the more accurate a context element describes a context at one point in time. Since the values are obtained by measurements, we may assume that the count of context elements is finite.

In [75] it was suggested that the context types location, identity, activity and time are more important than other types in order to describe a context. Undoubtedly, studies that utilise these context types for context-aware applications dominate studies on other context types. One reason for this is that implications obtained from these mentioned context types seem to be intuitive to most people. However, we argue that the type of context useful for an application is inherently dependent on the application type and that this context might be ignorant of the location, identity, activity or time.

Consider, for example, an arbitrary person sitting in her room and reading a book. While this scenario appears to be tranquil when only the four context types location, identity, activity and time are taken into account, the general assessment might change with the utilisation of further context sources. If, for example, the room temperature instantly rises or the amount of methane in the air increases, the same situation then appears in a different light. Danger might be at hand and a swift reaction is required.

We therefore assume that the application defines the relevance of distinct context types. The relevance could be modified by any kind of weighting or duplicating of contexts. Since we propose an architecture that utilises contexts for arbitrary applications, we do not prefer any context type above any other. For the remainder of this document we do not bother about the correct and application specific weighting, but assume that the contexts utilised have been filtered and weighted according to the application needs in advance. Several aspects of context have been introduced in [76, 77]. A further structured and extended distinction of context types is depicted in figure 2.2. This figure should be understood as a working model of context aspects. Context specifications for the context classes depicted in the figure are examples and can be carried on by other examples that logically fit into the corresponding context class. Further aspects of context not depicted in the figure might well be found.

### 2.2.4 Context abstraction levels

Context does not necessarily equal context. Two contexts of the same type that describe the same time interval might nonetheless differ from each other in value. Context has several levels of abstraction depending on the amount of pre-processing applied. A temperature context might, for example, hold the value $24°C$ as well as the value ‘warm ’. These context values might originate from identical measurements of context sources. However, the data abstraction level differs. The value ‘warm’ is at a higher abstraction level than the value $24°C$.

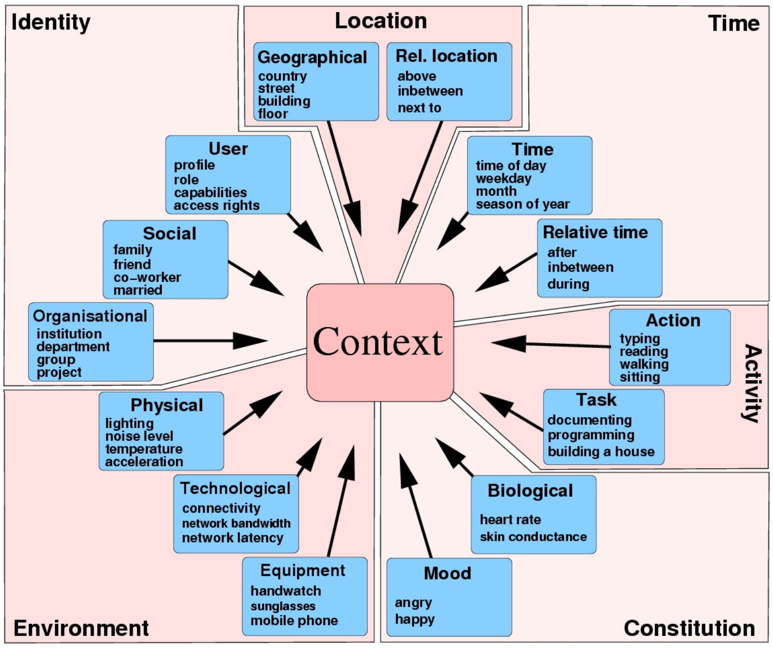


Figure 2.2: Aspects of context

Although several authors use the notions high-level context, low-level context and raw data, in order to describe various context abstraction levels, no exact definition of these notions is given in the literature. These notions are therefore often used with different meanings. Some authors, for example, use the term low-level context in the same sense as other authors use the term raw data. Typically, higher context representations tend to be symbolic while lower representations are more often numeric. Generally, the definition of several data abstraction levels is reasonable since the kind of representation used for operations on context elements may affect the accuracy of the operation [78].

A rough distinction between low-level and higher level contexts is made by Anind K. Dey, Bill Schilit and Marvin Theimer [58, 52]. Following this discussion, low-level context is used synonymously for data directly output from sensors, while high-level contexts are further processed. This processing can, for example, be an aggregation, an interpretation, a data calibration, noise removal or reforming of data distributions.

Jani Mäntyjärvi further distinguishes between processed contexts that describe an action or a condition [59]. Following his notion, raw data can be, for example, $24°C$ or 70% humidity. While for low-level contexts these are further processed to conditions like 'warm' or 'high humidity'. Finally, a high-level context is an activity as, for instance, 'having lunch'.

Actually, these distinctions between high-level and low-level contexts are only required (and properly understood) by humans. From a computational viewpoint, actions and conditions are both string values obtained by further processing of raw data. From a computation-centric standpoint, both constructs are consequently on the same level of data abstraction.

| High-level context | Low-level context | Raw data | Context source |
|---|---|---|---|
| walking | 14°C | 001001111 | thermometer |
| walking | 57.2°F | 001001111 | thermometer |
| watching movie | 64dB | 109 | microphone |
| listening music | 64dB | 109 | microphone |
| at the beach | 47° 25.5634’N; 007° 39.3538’E | GPRMC[3] | GPS sensor |
| swimming | 47° 25.5634’N; 007° 39.3538’E | GPGGA[4] | GPS sensor |
| writing | z | 0x79 | keyboard [en] |
| writing | ы | 0x79 | keyboard [ru] |
| writing | z | 0x7a | keyboard [de] |
| office occupied | z | 0x7a | keyboard [de] |

Table 2.1: High-level contexts, low-level contexts and raw context data for exemplary context sources.

[8] GPRMC Example:
$GPRMC,191410,A,4725.5634,N,00739.3538,E,0.0,0.0,181102,0.4,E,A*19

[9] GPGGA Example:
$GPGGA,191410,4725.5634,N,00739.3538,E,1,04,4.4,351.5,M,48.0,M,,*45

### A computation-centric approach

We therefore take an alternative, computation-centric, approach and classify the level of abstraction of contexts by the amount of pre-processing applied to the data. We distinguish between high-level context information, low-level context information and raw context data[2] (cf. table 2.1).

In table 2.1, exemplary raw context data, low-level contexts and high-level contexts are depicted. Note that in all data abstraction levels different context representations are possible even if the measurement is identical. An example well-suited to illustrate this is the keyboard sensor. The same key pressed on an English and a Russian keyboard (raw context data identical) might result in different low-level contexts due to an alternative language setting (acquisition procedure). In the Cyrillic layout the letter ’ы’ is obtained while it is the letter ’z’ for the English layout.

However, for German keyboards the letters ’y’ and ’z’ are exchanged compared to the English layout, hence leading to the same low-level context even though the raw context data is different. Furthermore, different context interpretation procedures may lead to

[2] For ease of presentation, we utilise the notions ’raw data’ and ’raw context data’ synonymously.

distinct high-level contexts (office occupied or writing).

A discussion of the three data abstraction levels ‘raw context data’, ‘low-level context’ and ‘high-level context’ is given in the following.

The output of any context source is considered as raw data since it most probably needs further interpretation. Already at the very first abstraction level of raw context data, basic operations on the measured samples might be suggestive. Computations that might be applied on this data include mechanisms to correct possible measurement or sensor errors, filters that might abstract from irrelevant measurements or also processes that weight the measurements.

Different manufacturers produce sensors with varying output even though the sensors might belong to the same class. This is because of possibly different encoding of the sensed information or due to a different representation or accuracy. Two temperature sensors may, for instance, differ in the unit (Celsius or Fahrenheit), in the measurement accuracy or in the measurement range. A pre-processing of raw context data is necessary so that further processing is not influenced by special properties of the context source itself. We refer to this pre-processing as the context acquisition step.

The data has become low-level context elements after the context acquisition. The low-level contexts of two arbitrary context sources of the same class measured at the same time in the same place is identical with the exception of a possibly differing measurement accuracy, provided that both context sources are in good order. The output of all context sources for temperature may, for example, be represented in degree Celsius.

In order to obtain high-level context elements, further processing operations are applied. Possible operations are aggregation, interpretation, semantic reasoning, data calibration, noise removal or reforming of data distributions. We refer to this pre-processing as the context interpretation step.

From low-level contexts describing the temperature, light intensity and the humidity it might be possible to infer the high-level context outdoors/indoors. There is no limit to the level of context interpretation. Several high-level contexts may be aggregated to again receive high-level context elements. For our discussion, however, we do not distinguish between high-level contexts of various context abstraction levels. For these three context abstraction levels, the distinguishing factor is the amount of pre-processing applied. Note, however, that we do not exactly define the amount of pre-processing for all three context abstraction levels since it may vary between distinct application scenarios. For our discussion it suffices that this construct of context abstraction levels is hierarchical. The amount of pre-processing applied to high-level contexts always exceeds the amount of pre-processing applied to low-level contexts in the same application scenario.

Observe that it is possible that two contexts of the same context type are differing in their context abstraction level when the amount of pre-processing to derive these contexts differs. While this might intuitively appear inconsistent, it is inherently logical from a computation-centric viewpoint. The amount of computation or pre-processing applied to contexts of distinct context abstraction levels differs. In addition, the information certitude of contexts in distinct abstraction levels might differ. Various context processing steps and corresponding input and output data are depicted in figure 2.3.

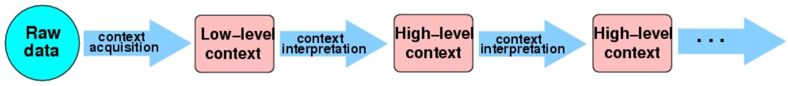


Figure 2.3: Context pre-processing steps.

### 2.2.5 Context data types

Since context is acquired from a set of heterogeneous context sources and is computed at various levels of abstraction, context processing operations applicable to one subset of contexts might be inapplicable to another subset.

As an example, consider IP addresses as context type on the one hand and temperature as another context type. Temperature contexts contain an implicit order regarding their magnitude while for IP addresses, an order cannot be provided in the same manner.

In [76] four data types have been introduced that group contexts applicable to the same mathematical operations together. Following this discussion, we distinguish context data types between nominal, ordinal and numerical categories. We omit the fourth category interval that was proposed in [76] since the boundaries of any context type (the only use for the interval category described in [76]) are provided for ordinal and numerical contexts in our case anyway.

The only operation applicable to nominal context data is the equals operation. Contexts of nominal context data type are, for example, arbitrary binary contexts, whereas symbolic context representations like, for instance, activities (walking, talking) or tasks (cleaning) are of nominal context data type.

Ordinal context data types further allow the test for an order between these contexts. Examples for contexts of ordinal context data type are physical contexts like lighting or acceleration when represented in symbolic notation like 'dark' and 'bright' or 'fast' and 'slow'.

Contexts of numerical context data type allow arbitrary mathematical operations to be applied on them. A good example for these context data types is the time. By subtraction, the time difference between two contexts of this type can be calculated.

We further consider hierarchical contexts, that are applicable to the 'subset'-operation. Similar to ordinal context data types, for hierarchical context data types an ordering of the contexts is possible. However, the order might be any kind of hierarchy as a directed tree or graph structure. Examples for a context type of this class are geographical contexts in a symbolic representation as 'in office building' or 'in town'.

The operators applicable to one context type limit the number of appropriate context processing methods. A context processing method usually requires a minimum set of operations on contexts. In order to be processed by a processing method, all processed contexts therefore have to share this minimum set of operations. An easy solution to equalise all contexts is to abstract from all operators not applicable to the whole set of available contexts. Clearly, this reduces the already sparse information we have about the data and artificially restricts us to a smaller number of context processing methods.

| Context type | nominal | ordinal | hierarchical | numerical |
|---|---|---|---|---|
| Organisational | + | | + | |
| Social | + | | + | |
| User | + | | | |
| Geographical | + | | + | |
| Relative location | + | | + | |
| Task | + | | + | |
| Action | + | | | |
| Time | + | + | + | + |
| Relative time | + | + | | |
| Biological | + | + | | |
| Mood | + | | | |
| Physical | + | + | | + |
| Technological | + | + | | + |
| Equipment | + | | + | |

Table 2.2: Operators applicable to various context types

Table 2.2 depicts the context data types of the context types introduced in figure 2.2[5].

Observe that the context data type is not related to the data abstraction level of contexts. Low-level and high-level contexts alike might be of ordinal, nominal, numeric or hierarchical context data type.

From one context abstraction level to the next higher one, the context data type may swap to an arbitrary other context data type. While, for instance, in the aggregation of contexts, the resulting context might likely support less operations than the operations applicable to the set of contexts before the aggregation, it is also feasible to add further operations by a mapping of contexts to elements that support these further operations.

### 2.2.6 Representation and illustration of contexts

We have now introduced the concept of context and have discussed context types, context abstraction levels and context data types at a rather abstract, theoretical level. For any problem domain, a good perception of the contexts and relations in this domain is at least helpful for the next step, the approach to solve the problem at hand.

A straightforward way to illustrate low-level contexts is to map them into a multi-

[5]The classification of context types to context data types represents one example classification that is considered reasonable by the authors. However, a specific scenario might introduce context type classifications that differ from the values depicted in the table. The important point here is that in a given scenario the observed context data types might not be computed by arbitrary context prediction algorithms

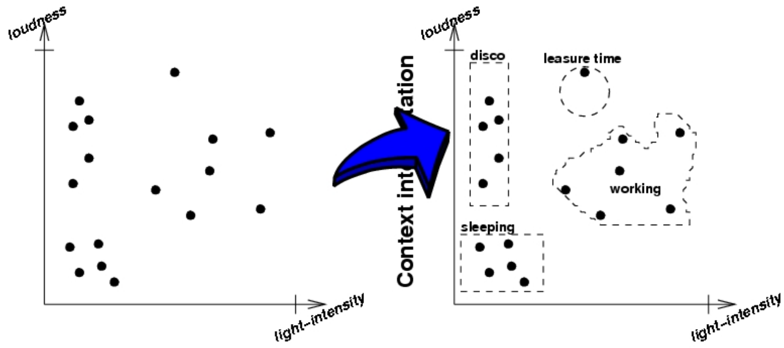


Figure 2.4: Illustration of the context interpretation step.

dimensional coordinate system. This representation has first been considered by Padovitz et al [79, 80, 81]. Although another distinction between low-level contexts and high-level contexts has been applied, the same principle can also be applied in our case. The general idea is to represent for every time interval a low-level context element by a vector in a multi-dimensional coordinate system. Each coordinate axis represents a normalised aspect of a low-level context element.

High-level contexts are then sets of low-level contexts that are assigned a label. Figure 2.4 illustrates the context interpretation step[6].

Low-level contexts are represented on the left hand side by dots in a coordinate system. On the right hand side, these low-level contexts are transformed to high-level contexts which is basically a grouping of several low-level contexts together into a set of high-level contexts.

This geometrical context representation is trivially extended to context sequences in time by simply considering one further axis in the coordinate system that represents the time. This more concrete, geometrical context representation assists us in the discussion of several properties later on.

In our discussion we do not consider overlapping high-level definitions. A discussion of this topic can be found in [81].

[6] The figure connotes that the high-level contexts 'sleeping', 'working', 'leasure time' and 'disco' can be distinguished by the light intensity and the loudness. This labelling of high-level contexts is only for an easier understanding of the described context interpretation step. Note that the necessary context sources to accurately distinguish the mentioned high-level contexts is currently an unsolved research problem.

# Index

# List of Tables

# List of Figures

# Bibliography


[1] Culler, D., Estrin, D., Srivastava, M.: Overview of sensor networks. IEEE Computer **37**(8) (2004) 41–49

[2] Zhao, F., Guibas, L.: Wireless Sensor Networks: An Information Processing Approach. Morgan Kaufmann, Los Altos, CA (2004)

[3] Norman, D.: The invisible computer. MIT press (1999)

[4] Butera, W.J.: Programming a paintable computer. PhD thesis, Massachusetts Institute of Technology (2002)

[5] Pillutla, L., Krishnamurthy, V.: Joint rate and cluster optimisation in cooperative mimo sensor networks. In: Proceedings of the 6th IEEE Workshop on signal Processing Advances in Wireless Communications. (2005) 265–269

[6] Scaglione, A., Hong, Y.W.: Opportunistic large arrays: Cooperative transmission in wireless multihop ad hoc networks to reach far distances. IEEE Transactions on Signal Processing **51**(8) (2003) 2082–2092

[7] Sendonaris, A., Erkop, E., Aazhang, B.: Increasing uplink capacity via user cooperation diversity. In: IEEE Proceedins of the International Symposium on Information Theory (ISIT). (2001) 156

[8] Laneman, J., Wornell, G., Tse, D.: An efficient protocol for realising cooperative diversity in wireless networks. In: Proceedings of the IEEE International Symposium on Information Theory. (2001) 294

[9] Hong, Y.W., Scaglione, A.: Critical power for connectivity with cooperative transmission in wireless ad hoc sensor networks. In: IEEE Workshop on Statistical Signal Processing. (2003)

[10] Hong, Y.W., Scaglione, A.: Energy-efficient broadcasting with cooperative transmission in wireless sensor networks. IEEE Transactions on Wireless communications (2005)

[11] Jayaweera, S.K.: Energy analysis of mimo techniques in wireless sensor networks. In: 38th conference on information sciences and systems. (2004)

[12] del Coso, A., Sagnolini, U., Ibars, C.: Cooperative distributed mimo channels in wireless sensor networks. IEEE Journal on Selected Areas in Communications **25**(2) (2007) 402–414

[13] Sigg, S., Beigl, M.: Collaborative transmission in wsns by a (1+1)-ea. In: Proceedings of the 8th International Workshop on Applications and Services in Wireless Networks (ASWN'08). (2008)

[14] Sigg, S., Beigl, M.: Randomised collaborative transmission of smart objects. In: 2nd International Workshop on Design and Integration principles for smart objects (DIPSO2008) in conjunction with Ubicomp 2008. (2008)

[15] Mudumbai, R., Brown, D.R., Madhow, U., Poor, H.V.: Distributed transmit beamforming: Challenges and recent progress. IEEE Communications Magazine (2009) 102–110

[16] Mudumbai, R., Wild, B., Madhow, U., Ramchandran, K.: Distributed beamforming using 1 bit feedback: from concept to realization. In: Proceedings of the 44th Allerton conference on communication, control and computation. (2006) 1020–1027

[17] Barriac, G., Mudumbai, R., Madhow, U.: Distributed beamforming for information transfer in sensor networks. In: Proceedings of the third International Workshop on Information Processing in Sensor Networks. (2004)

[18] Mudumbai, R., Barriac, G., Madhow, U.: On the feasibility of distributed beamforming in wireless networks. IEEE Transactions on Wireless communications **6** (2007) 1754–1763

[19] Ochiai, H., Mitran, P., Poor, H.V., Tarokh, V.: Collaborative beamforming for distributed wireless ad hoc sensor networks. IEEE Transactions on Signal Processing **53**(11) (2005) 4110 – 4124

[20] Chen, W., Yuan, Y., Xu, C., Liu, K., Yang, Z.: Virtual mimo protocol based on clustering for wireless sensor networks. In: Proceedings of the 10th IEEE Symposium on Computers and Commmunications. (2005)

[21] Youssef, M., Yousif, A., El-Sheimy, N., Noureldin, A.: A novel earthquake warning system based on virtual mimo wireless sensor netwroks. In: Canadian conference on electrical and computer engineering. (2007) 932–935

[22] del Coso, A., Savazzi, S., Spagnolini, U., Ibars, C.: Virtual mimo channels in cooperative multi-hop wireless sensor networks. In: 40th annual conference on information sciences and systems. (2006) 75–80

[23] Jayaweera, S.K.: Energy efficient virtual mimo based cooperative communications for wireless sensor networks. IEEE Transactions on Wireless communications **5**(5) (2006) 984–989

[24] Laneman, J., Wornell, G.: Distributed space-time coded protocols for exploiting cooperative diversity in wireless networks. IEEE Transactions on Information theory **49**(10) (2003) 2415–2425

[25] Sendonaris, A., Erkip, E., Aazhang, B.: User cooperation diversity – part i: System description. IEEE Transactions on Communications **51**(11) (2003) 1927–1938

[26] Zimmermann, E., Herhold, P., Fettweis, G.: On the performance of cooperative relaying protocols in wireless networks. European Transactions on Telecommunications **16**(1) (2005) 5–16

[27] Cover, T.M., Gamal, A.A.E.: Capacity theorems for the relay channel. IEEE Transactions on Information Theory **525**(5) (1979) 572–584

[28] Kramer, G., Gastpar, M., Gupta, P.: Cooperative strategies and capacity theorems for relay networks. IEEE Transactions on Information Theory **51**(9) (2005) 3037–3063

[29] Scaglione, A., Hong, Y.W.: Cooperative models for synchronization, scheduling and transmission in large scale sensor networks: An overview. In: 1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing. (2005) 60–63

[30] Gupta, P., Kumar, R.P.: The capacity of wireless networks. IEEE Transactions on Information Theory **46**(2) (2000) 388–404

[31] Mitran, P., Ochiai, H., Tarokh, V.: Space-time diversity enhancements using collaborative communications. IEEE Transactions on Information Theory **51**(6) (2005) 2041–2057

[32] Simeone, O., Spagnolini, U.: Capacity region of wireless ad hoc networks using opportunistic collaborative communications. In: Proceedings of the International Conference on Communications (ICC). (2006)

[33] Krohn, A., Beigl, M., Decker, C., Varona, D.G.: Increasing connectivity in wireless sensor network using cooperative transmission. In: 3rd International Conference on Networked Sensing Systems (INSS). (2006)

[34] Krohn, A.: Optimal non-coherent m-ary energy shift keying for cooperative transmission in sensor networks. In: 31st IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). (2006)

[35] Hong, Y.W., Scaglione, A.: Cooperative transmission in wireless multi-hop ad hoc networks using opportunistic large arrays (ola). In: SPAWC. (2003)

[36] Brown, D.R., Prince, G., McNeill, J.: A method for carrier frequency and phase synchronization of two autonomous cooperative transmitters. In: sixth IEEE workshop on signal processing advances in wireless communications. (2005)

[37] Brown, D.R., Poor, H.V.: Time-slotted round-trip carrier synchronisation for distributed beamforming. IEEE Transactions on Signal Processing **56** (2008) 5630–5643

[38] Ozil, I., Brown, D.R.: Time-slotted round-trip carrier synchronisation. In: Proceedings of the 41st Asilomar conference on signals, signals and computers. (2007) 1781–1785

[39] Tu, Y., Pottie, G.: Coherent cooperative transmission from multiple adjacent antennas to a distant stationary antenna through awgn channels. In: Proceedings of the IEEE Vehicular Technology Conference. (2002) 130–134

[40] Mudumbai, R., Hespanha, J., Madhow, U., Barriac, G.: Scalable feedback control for distributed beamforming in sensor networks. In: Proceedings of the IEEE International Symposium on Information Theory. (2005) 137–141

[41] Mudumbai, R., Hespanha, J., Madhow, U., Barriac, G.: Distributed transmit beamforming using feedback control. IEEE Transactions on Information Theory ((In review))

[42] Seo, M., Rodwell, M., Madhow, U.: A feedback-based distributed phased array technique and its application to 60-ghz wireless sensor network. In: IEEE MTT-S International Microwave Symposium Digest. (2008) 683–686

[43] Bucklew, J.A., Sethares, W.A.: Convergence of a class of decentralised beamforming algorithms. IEEE Transactions on Signal Processing **56**(6) (2008) 2280–2288

[44] Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste, P.: Project aura: Toward distraction-free pervasive computing. IEEE Pervasive computing **4** (2002) 22–31

[45] Want, R., Schilit, B., Adams, N., Gold, R., Petersen, K., Goldberg, D., Ellis, J., Weiser, M.: An overview of the parctab ubiquitous computing experiment. In: IEEE personal communications. Volume 2. (1995) 28–43

[46] Gellersen, H.W., Beigl, M., Krull, H.: The mediacup: Awareness technology embedded in an everyday object. In Gellersen, H.W., ed.: 1th International Symposium on Handheld and Ubiquitous Computing (HUC99). Volume 1707 of Lecture notes in computer science., Springer (1999) 308–310

[47] Capra, L., Musolesi, M.: Autonomic trust prediction for pervasive computing. In: Proceedings of IEEE Workshop on Trusted and Autonomic Computing Systems 2006 (TACS'06). (2006)

[48] Ferscha, A., Holzman, C., Leitner, M.: Interfaces everywhere – interacting with the pervasive computer (2006) Half-day tutorial, 10th ACM International Conference on Intelligent User Interfaces (IUI 2006), Sydney, Australia.

[49] Want, R., Hopper, A., Falcao, V., Gibbons, J.: The active badge location system. In: ACM Transactions on Information Systems. Volume 1. (1992) 91–102

[50] Weiser, M.: The computer for the 21st century. In: Scientific American. Volume 3. (1991) 66–75

[51] Dourish, P.: What we talk about when we talk about context. In: Personal and Ubiquitous Computing. Volume 8. (2004)

[52] Schilit, B.N., Theimer, M.M.: Disseminating active map information to mobile hosts. In: IEEE Network. Volume 5. (1994) 22–32

[53] Schilit, B.N., Adams, N., Want, R.: Context-aware computing applications. In: IEEE Workshop on Mobile Computing Systems and Applications. (1994)

[54] Brown, P.J., Bovey, J.D., Chen, X.: Context-aware applications: from the laboratory to the marketplace. In: IEEE personal communications. Volume 4. (1997) 58–64

[55] Pascoe, J.: The stick-e note architecture: Extending the interface beyond the user. In: Proceedings of the 2nd International Conference on Intelligent user Interfaces. (1997) 261–264

[56] Dey, A.K., Abowd, G.D., Wood, A.: Cyberdesk: A framework for providing self-integrating context-aware services. In: Knowledge-Based Systems. Volume 11. (1998) 3–13

[57] Schmidt, A., Beigl, M.: There is more to context than location: Environment sensing technologies for adaptive mobile user interfaces. In: Workshop on Interactive Applications of Mobile Computing (IMC'98). (1998)

[58] Dey, A.K.: Providing architectural support for building context–aware applications. PhD thesis, Georgia Institute of Technology (2000)

[59] Mäntyjärvi, J.: Sensor-based context recognition for mobile applications. PhD thesis, VTT Technical Research Centre of Finland (2003)

[60] Henricksen, K.: A Framework for Cotnext-Aware Pervasive Computing Applications. PhD thesis, School of Information Technology and Electrical Engineering at the University of Queensland (2003)

[61] Lieberman, H., Selker, T.: Out of context: Computer systems that adapt to, and learn from, context. In: IBM Systems Journal. Volume 39. (2000) 617–632

[62] Fitzpatrick, A., Biegel, G., Cahill, S.C.V.: Towards a sentient object model. In: Workshop on Engineering Context-Aware Object Oriented Systems and Environments (ECOOSE). (2002)

[63] Pascoe, J.: Adding generic contextual capabilities to wearable computers. In: Proceedings of the second International Symposium on Wearable Computers. (1998) 92–99

[64] Dey, A.K., Salber, D., Abowd, G.D., Futakawa, M.: The conference assistant: comtining context-awareness with wearable computing. In: Proceedings of the third International Symposium on Wearable Computers. (1999) 21–28

[65] Kortuem, G., Segall, Z., Bauer, M.: Context-aware, adaptive wearable computers as remote interfaces to 'intelligent' environments. In: Proceedings of the second International Symposium on Wearable Computers. (1998) 58–65

[66] Chen, G.: Solar: Building A Context Fusion Network for Pervasive Computing. PhD thesis, Hanover, New Hampshire (2004)

[67] Schmidt, A.: Ubiquitous Computing – Computing in Context. PhD thesis, Lancaster University, UK (2002)

[68] Himberg, J.: From insights to innovations: data mining, visualisation, and user interfaces. PhD thesis, Helsinki University of Technology (2004)

[69] Himberg, J., Korpiaho, K., Mannila, H., Tikanmäki, J., Toivonen, H.: Time series segmentation for context recognition in mobile devices. In: Proceedings of the 2001 IEEE International Conference on Data Mining. (2001) 203–210

[70] Mäntyjärvi, J., Himberg, J., Huuskonen, P.: Collaborative context recognition for handheld devices. In: Proceedings of the first IEEE International Conference on Pervasive Computing and Communications (PerCom'03). (2003) 161–168

[71] Schilit, W.N.: A System Architecture for Context-Aware Mobile Computing. PhD thesis, Columbia University (1995)

[72] Dey, A.K., Abowd, G.D., Salber, D.: A context-based infrastructure for smart environments. In: Proceedings of the first International Workshop on Managing Interactions in Smart Environments (MANSE'99). (1999) 114–128

[73] Schmidt, A., Laerhoven, K.V., Strohbach, M., Friday, A., Gellersen, H.W.: Context acquisition based on load sensing. In: Proceedings of Ubicomp 2002, Lecture Notes in Computer Science. Volume 2498., Springer Verlag (2002) 333 – 351

[74] Jacob, R.J., Ishii, H., Pangaro, G., Patten, J.: A tangible interface for organising information using a grid. In: Conference on Human Factors in Computing Systems (CHI 2002). (2002)

[75] Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing, London, UK, Springer-Verlag (1999) 304–307

[76] Mayrhofer, R.M.: An Architecture for Context Prediction. PhD thesis, Johannes Kepeler University of Linz, Altenbergstrasse 69, 4040 Linz, Austria (2004)

[77] Mayrhofer, R.M., Radi, H., Ferscha, A.: Recognising and predicting context by learning from user behaviour. In: The International Conference On Advances in Mobile Multimedia (MoMM2003). Volume 171. (2003) 25–35

[78] Brooks, R.A.: Elephants don't play chess. In: Robotics and Autonomous Systems. Volume 6. (1990)

[79] Padovitz, A., Bartolini, C., Zaslavski, A., Loke, S.W.: Extending the context space approach to management by business objectives. In: 12th Workshop of the HP OpenView University Association. (2005)

[80] Padovitz, A., Loke, W.W., Zaslavsky, A.: On uncertainty in context-aware computing: Appealing to high-level and same-level context for low-level context verification. In: Proceedings of the 1st International Workshop on Ubiquitous Computing (IWUC'04). (2004) 62–72

[81] Padovitz, A., Loke, S.W., Zaslavsky, A., Burg, B.: Towards a general approach for reasoning about context, situations and uncertainty in ubiquitous sensing: Putting geometrical intuitions to work. In: 2nd International Symposium on Ubiquitous Computing Systems (UCS'04). (2004)

[82] Li, Y., Thai, M., Wu, W.: Wireless sensor networks and applications. Signals and Communication Technology. Springer (2008)

[83] Karl, H., Willig, A.: Protocols and Architectures for Wireless Sensor Networks. Wiley (2005)

[84] Rappaport, T.: Wireless Communications: Principles and Practice. Prentice Hall (2002)

[85] Sohrabi, K., Manriquez, B., Pottie, G.: Near-ground wideband channel measurements. In: Proceedings of the 49thvehicular technology conference. (1999) 571–574

[86] Kahn, J.M., Katz, R.H., Pister, K.S.J.: Next century challenges: Mobile networking for smart dust. In: Proceedings of the ACM MobiCom. (1999) 271–278

[87] Hoblos, G., Staroswiecki, M., Aitouche, A.: Optimal design of fault tolerant sensor networks. In: Proceedings of the IEEE International Conference on Control Applications. (2000) 467–472

[88] Bulusu, N., Estrin, D., Girod, L., Heidemann, J.: Scalable coordination for wireless sensor netwroks: Self-configuring localisation systems. In: Proceedings of the 6th IEEE International Symposium on Communication Theory and Application. (2001)

[89] Shen, C.C., Srisathapornphat, C., Jaikaeo, C.: Sensor information networking architecture and applications. IEEE Personal Communications **8**(4) (2001) 52–59

[90] Cerpa, A., Elson, J., Estrin, D., Girod, L., Hamilton, M., Zhao, J.: Habitat monitoring: Application driver for wireless communications technology. In: Proceedings of the ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean. (2001)

[91] of IEEE 802.11, T.E.: Ieee standard for wireless lan medium access control (mac) and physical layer (phy) specifications (1997)

[92] Jerome, J.K.: Three men in a boat. Collector's Library (2005)

[93] Seybold, J.S.: Introduction to RF propagation. Wiley (2005)

[94] 3GPP: 3rd generation partnership project; technical specification group radio access networks; 3g home nodeb study item technical report (release 8). Technical Report 3GPP TR 25.820 V8.0.0 (2008-03) (March)

[95] Ohm, J.R., Lüke, H.D.: Signalübertragung – Grundlagen der digitalen und analogen Nachrichtenübertragungssysteme. Volume 10. Springer (2007)

[96] Feller, W.: An Introduction to Probability Theory and its Applications. Wiley (1968)

[97] Duda, R., Hart, P., Stork, D.: Pattern Classification. 2nd edn. Wiley Interscience (2001)

[98] Schwefel, H.P.: Direct search for optimal parameters within simulation models. Proceedings of the twelfth annual simulation symposium (1979) 91–102

[99] Holland, J.: Genetic algorithms and the optimal allocation of trials. SIAM, Journal of computing **3**(4) (1974) 88–105

[100] Schwefel, H.P.: Evolution and optimum seeking. Wiley-Interscience (1995)

[101] Fogel, L.J., Fogel, D.B.: A preliminary investigation on extending evolutionary programming to include self-adaptation on finite state. Informatica **18**(4) (1994)

[102] Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press (1992)

[103] Bäck, T.: An overview of parameter control methods by self-adaptation in evolutionary algorithms. Fundamenta Informaticae **35** (1998) 51–66

[104] Droste, S.: Efficient genetic programming for finding good generalizing boolean functions. In Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L., eds.: Proceedings of the second Genetic Programming conference (GP 97), Morgan Kaufmann (1997) 82–87

[105] Droste, S., Heutelbeck, D., Wegener, I.: Distributed hybrid genetic programming for learning boolean functions. In Schoenauer, M., ed.: Proceedings of the 6th Parallel Problem Solving from Nature (PPSN VI). Volume 1917 of Lecture Notes in Computer Science (LNCS)., Springer (2000) 181–190

[106] Jansen, T., Wegener, I.: Evolutionary algorithms – how to cope with plateaus of constant fitness and when to reject strings of the same fitness. IEEE Transactions on Evolutionary Computation **5**(6) (2001) 589–599

[107] Wegener, I., Witt, C.: On the optimization of monotone polynomials by the (1+1)ea and randomized local search. In: Genetic and Evolutionary Computation Conference (GECCO). Number 2723 in Lecture notes in computer sciences (LNCS) (2005) 622–633

[108] Wegener, I., Witt, C.: On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions. Journal of Discrete Algorithms (3) (2005) 61–78

[109] Droste, S., Jansen, T., Wegener, I.: A rigorous complexity analysis of the (1+1) evolutionary algorithm for linear functions with boolean inputs. In: Proceedings of the third IEEE International conference on Evolutionary computation (ICEC 98), Piscataway, NJ, IEEE press (1998) 499–504

[110] Droste, S., Jansen, T., Wegener, I.: A rigorous complexity analysis of the (1+1) evolutionary algorithm for separable functions with boolean inputs. Evolutionary Computation **6**(2) (1998) 185–196

[111] Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science **276**(1-2) (2002) 51 – 81

[112] Sirkeci-Mergen, B., Scaglione, A.: 10. In: Randomized cooperative transmission in large scale sensor networks. A. Swami, Q. Zhao, Y. Hong, and L. Tong, Wiley (2007)

[113] Laneman, J., Wornell, G.: Energy-efficient antenna sharing and relaying for wireless networks. In: Proceedings of the IEEE Wireless Communication Networking Conference. (2000) 294

[114] Lanemann, J., Tse, D., Wornell, G.: Cooperative diversity in wireless networks: Efficient protocols and outage behaviour. Transactions on inforamtion Theory **50**(12) (2004)

[115] Nabar, R., Bölscskei, H., Wornell, G.: Fading relay channels: Performance limits and space-time signal design. IEEE Journal on Selected Areas in Communications **22**(6) (2004) 1099–1109

[116] Gastpar, M., Vetterli, M.: On the capacity of large gaussian relay networks. IEEE Transactions on Information Theory **51**(3) (2005) 765–779

[117] Ahlswede, R., Cai, N., Li, S.Y., Yeung, R.: Network information flow. IEEE Transactions on Information Theory **46**(4) (2000) 1204–1216

[118] Li, S.Y., Son, S., Stankovic, J.: Linear network coding. IEEE Transactions on Information Theory **49**(2) (2003) 371–381

[119] Woldegebreal, D.H., Karl, H. In: Network-Coding-Based Cooperative Transmission in Wireless Sensor Networks: Diversity-Multiplexing Tradeoff and Coverage Area Extension. Volume 4913 of Lecture Notes in Computer Science (LNCS). Springer-Verlag (2008) 141–155

[120] Scaglione, A., Hong, Y.W.: Opportunistic large arrays. In: IEEE International Symposium on Advances in Wireless Communications. (2002)

[121] Salhotra, A., Scaglione, A.: Multiple access in connectionless networks using cooperative transmission. In: Allerton Conference. (2003)

[122] Paulraj, A., Nabar, R., Gore, D.: Introduction to Space-Time Wireless communications. Cambridge University Press (2003)

[123] Shuguang, C., Goldsmith, A.: Energy-efficiency of mimo and cooperative mimo techniques in sensor networks. IEEE Journal on Selected Areas in Communications **22**(6) (2004) 1089–1098

[124] Alamouti, S.M.: A simple transmit diversity technique for wireless communications. IEEE Journal on select areas in communications **16**(8) (1998)

[125] Gastpar, M., Vetterli, M.: On the capacity of wireless networks: the relay case. In: Proceedings of the IEEE Infocom. (2002) 1577–1586

[126] Hagmann, W.: Network synchronisation techniques for satellite communication systems. PhD thesis, USC, Los Angeles (1981)

[127] Best, R.: Phase-Locked Loops: Design, Simulation and Applications. McGraw-Hill (2003)

[128] Krohn, A.: Superimposed Radio Signals for Wireless Sensor Networks. PhD thesis, Technical University of Braunschweig (2007)

[129] Bennett, W.: Introduction to signal transmission. McGraw-Hill (1971)

[130] Sigg, S., Masri, R.M.E., Beigl, M.: A sharp asymptotic bound for feedback based closed-loop distributed adaptive beamforming in wireless sensor networks. Transactions on mobile computing (2009 (submitted))