# Distributed Event Detection in Wireless Sensor Networks

Inauguraldissertation
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

**Markus Wälchli**

von Seeberg

Leiter der Arbeit:
Professor Dr. Torsten Braun
Institut für Informatik und angewandte Mathematik

Von der Philosophisch-naturwissenschaftlichen Fakultät angenommen.

Bern, 26.05.2009

Der Dekan
Prof. Dr. Urs Feller

## Abstract

In comparison to traditional sensor networks, wireless sensor networks have a number of strengths such as distributed operation, parallelism, redundancy, and comparatively high cost-effectiveness due to lack of wires. On the other hand, their tininess, need for long-term operation, and dependency on batteries impose severe restrictions on the system. Hence, services provided in sensor networks need to be lightweight in terms of memory and processing power and should not require high communication costs. In our own work we have developed an event monitoring architecture that provides energy-efficient medium access and topology control on the lower layers. On the application layer functionality to detect, track and classify occurring events in a lightweight and distributed manner is provided. The developed system has been used in an office access monitoring application, where illegal office access has been detected and reported.

# Contents


**1 Introduction** **1**
- 1.1 Problem Statement . . . 2
- 1.2 System Architecture . . . 4
- 1.3 System Evaluation . . . 5
- 1.4 Medium Access and Routing . . . 6
- 1.5 Event Detection and Tracking . . . 6
- 1.6 Event Localization and Signal Strength Estimation . . . 8
- 1.7 Event Classification . . . 9
- 1.8 Anomaly Detection . . . 9
- 1.9 Office Monitoring Application . . . 10
- 1.10 Contributions . . . 11
- 1.11 Thesis Outline . . . 11

**2 Wireless Sensor Networks** **13**
- 2.1 Introduction . . . 13
- 2.2 Sensing Capabilities and Sensors . . . 15
- 2.3 Factors Influencing Sensor Network Design . . . 16
  - 2.3.1 System Issues . . . 16
  - 2.3.2 Networking Issues . . . 18
  - 2.3.3 Environment . . . 19
- 2.4 Communication in Wireless Sensor Networks . . . 20
  - 2.4.1 Requirements . . . 20
  - 2.4.2 Multi-Hop Communication . . . 21
  - 2.4.3 Medium Access . . . 21
  - 2.4.4 Networking Layer . . . 22
  - 2.4.5 Application Layer . . . 24
- 2.5 Conclusions . . . 26

**3 Related Work** **27**
- 3.1 Introduction . . . 27
- 3.2 Medium Access Control . . . 28
  - 3.2.1 Synchronized Contention-Based Medium Access . . . 28
  - 3.2.2 Asynchronous Contention-Based Medium Access . . . 32
  - 3.2.3 Time Division Multiple Access . . . 33

3.3 Routing and Topology Control . . . . . 35
3.3.1 Routing Protocols for Wireless Sensor Networks . . . . . 35
3.3.2 Topology Control in Wireless Sensor Networks . . . . . 35
3.3.3 Connected Dominating Sets . . . . . 36
3.3.4 Multipoint Relaying Protocol . . . . . 39
3.4 Event Detection and Tracking . . . . . 40
3.4.1 Target Localization in Distributed Sensor Networks . . . . 41
3.4.2 SensIt: Region-Based Detection and Tracking of Targets . 41
3.4.3 Event Detection and Tracking with Consensus . . . . . 42
3.4.4 Event Detection Using Data Service Middleware . . . . . 43
3.4.5 EnviroTrack: An Environmental Computing Paradigm . . 43
3.4.6 Information-Driven Sensor Querying . . . . . 45
3.4.7 Algorithms for Fault-Tolerant Event Region Detection . . 45
3.5 Event Localization and Signal Strength Estimation . . . . . 45
3.5.1 Problem Formulation . . . . . 46
3.5.2 Simplex Downhill . . . . . 47
3.5.3 Conjugate Gradient Method . . . . . 48
3.5.4 Linear Least Square Method . . . . . 49
3.5.5 Tracking a Moving Object with a Binary Sensor Network . 49
3.5.6 PinPtr: Sensor Network-Based Countersniper System . . . 50
3.5.7 Localization with Positive and Negative Constraints . . . 51
3.5.8 SensIt: Distributed Localization and Signal Processing . . 51
3.5.9 Robust Localization of Multiple Events . . . . . 52
3.5.10 Distributed Optimization in Sensor Networks . . . . . 52
3.5.11 Distributed State Representation for Tracking Problems . . 53
3.5.12 Classical Node Positioning Methods . . . . . 53
3.6 Classification and Reasoning . . . . . 55
3.6.1 Learning Event Classes . . . . . 56
3.6.2 Bayesian Classifier . . . . . 57
3.6.3 Fuzzy Logic Controller . . . . . 58
3.6.4 Feedforward Neural Networks . . . . . 60
3.6.5 Adaptive Resonance Theory . . . . . 61
3.6.6 Haar Wavelet Transform . . . . . 63
3.6.7 Classification of Time-Discrete Events . . . . . 64
3.6.8 Continuous Event Classification and Anomaly Detection . 66
3.6.9 Threshold-based Event Classification . . . . . 69
3.7 Monitoring Applications . . . . . 70
3.7.1 Environmental Monitoring . . . . . 70
3.7.2 Visual Sensor Networks . . . . . 71
3.8 Sensor Node Platforms . . . . . 71
3.8.1 The Embedded Sensor Board . . . . . 71
3.8.2 The TmoteSky Platform . . . . . 73
3.9 Conclusions . . . . . 74

**4 Local Clock Synchronization** **75**
4.1 Introduction . . . 75
4.2 Local Adaptive Clock Assimilation Scheme (LACAS) . . . 77
4.3 Evaluation . . . 79
4.3.1 Simulation Scenario and Parameters . . . 79
4.3.2 Convergence of Schedule Length with LACAS . . . 80
4.3.3 Analysis of Power Consumption . . . 82
4.4 Conclusions . . . 84

**5 Backbone Support** **85**
5.1 Introduction . . . 85
5.2 Routing Backbone on the MAC Layer . . . 87
5.2.1 Long Sleep on the MAC . . . 88
5.2.2 CDS Construction Based on Multipoint Relaying . . . 89
5.2.3 Negotiation-Based CDS . . . 90
5.2.4 Reliability and Backbone Reconstruction . . . 93
5.3 Routing Backbone on the Network Layer . . . 94
5.3.1 Receiver-based Backbone Construction . . . 94
5.4 Evaluation . . . 99
5.4.1 Simulations on the MAC layer . . . 99
5.4.2 Simulations on the Network Layer . . . 103
5.4.3 Real-World Experiments . . . 109
5.5 Conclusions . . . 112

**6 Distributed Event Detection and Tracking** **115**
6.1 Introduction . . . 115
6.2 Distributed Group Formation and Maintenance . . . 117
6.2.1 Architecture Overview . . . 118
6.2.2 Leader Election and Maintenance . . . 119
6.3 Implementation Details . . . 120
6.4 Evaluation . . . 122
6.4.1 Simulations . . . 122
6.4.2 Real-World Experiments . . . 125
6.5 Conclusions . . . 128

**7 Event Localization and Signal Strength Estimation** **129**
7.1 Introduction . . . 129
7.2 Event-Based Localization and Signal Strength Estimation . . . 130
7.3 Evaluation . . . 130
7.3.1 Simulations . . . 131
7.3.2 Real-World Experiments . . . 133
7.4 Conclusions . . . 138

**8 Event Classification and Reasoning** **139**
8.1 Introduction . . . 139
8.2 Classification and Filtering of Time-Discrete Events . . . 141
8.2.1 Introduction . . . 141
8.2.2 Fuzzy Logic Controller . . . 142
8.2.3 Classifier Configuration . . . 145
8.3 Local Signal Processing with ART Neural Networks . . . 146
8.3.1 Introduction . . . 146
8.3.2 Local Fuzzy ART Neural Network . . . 147
8.4 Evaluation . . . 149
8.4.1 Classification of Time-Discrete Events . . . 149
8.4.2 Anomaly Detection and Signal Processing on Node Level 151
8.5 Conclusions . . . 155

**9 Office Monitoring Application** **157**
9.1 Introduction . . . 157
9.2 Office Monitoring . . . 158
9.2.1 System Design . . . 159
9.2.2 System-wide Anomaly Detection . . . 161
9.3 Anomaly Detection Performance . . . 163
9.3.1 Office Occupancy Patterns . . . 163
9.3.2 Computation at Desktop PC . . . 164
9.3.3 Detection Performance of the Anomaly Detectors . . . 164
9.3.4 Message Load . . . 165
9.3.5 Reporting and Triggering Delay . . . 166
9.3.6 Standalone ART Neural Network Performance . . . 167
9.4 Conclusions . . . 167

**10 Conclusions** **169**

**11 Future Work** **173**

**12 Acronyms** **175**

# List of Figures


1.1 General system architecture. . . . . 4
1.2 Typical deployment scenario. . . . . 5

2.1 Wireless sensor network with in-network processing. . . . . 14
2.2 Network stack implemented by sensor nodes. . . . . 20

3.1 Adaptive listening in T-MAC. . . . . 29
3.2 Drawback of virtual clustering. . . . . 31
3.3 Preamble sampling and data transmission. . . . . 32
3.4 Examples of a DS and a CDS. . . . . 37
(a) Dominating Set (DS). . . . . 37
(b) Connected Dominating Set (CDS). . . . . 37
3.5 CDS with pruning rules. . . . . 37
3.6 Connecting a maximum independent set. . . . . 38
3.7 Grid creation and handover in SensIt. . . . . 42
3.8 Group organization in EnviroTrack. . . . . 44
3.9 Simplex Downhill function minimization. . . . . 47
3.10 Geometrical operations of Simplex Downhill. . . . . 48
3.11 Conjugate Gradient and Steepest Descent. . . . . 48
3.12 Decomposition of multi-target tracking. . . . . 53
3.13 Membership functions used in fuzzy logics. . . . . 59
3.14 ART neural network architecture. . . . . 61
3.15 Haar Wavelet transform (LPF). . . . . 63
3.16 ESB sensor node. . . . . 72
3.17 Output of the TAOS TSL245 infrared to frequency converter [55]. 73
3.18 TmoteSky sensor node. . . . . 73

4.1 Periodic sleeping and virtual clustering. . . . . 76
4.2 Gravitation principle of LACAS: Cluster detection and merging. . 78
4.3 Schedule length convergence in a network consisting of 50 nodes. 80
(a) Evolution in the first hour (log-scaled). . . . . 80
(b) Evolution ignoring the first 100 s. . . . . 80
4.4 Schedule length evolution in larger networks (log-scaled). . . . . 81
(a) Network consisting of 100 nodes. . . . . 81
(b) Network consisting of 200 nodes. . . . . 81

4.5 Ripple effect of gravitation over multiple hops. . . . . 81

5.1 Routing backbone in a sensor network. . . . . 86
5.2 Neighborhood discovery and backbone construction. . . . . 87
5.3 Operation of backbone node (B) and redundant node (R). . . . . 88
5.4 Example of dominator election with MPR-based CDS. . . . . 90
5.5 Example of dominator election with N-CDS. . . . . 92
5.6 Example of dominator election with R-CDS. . . . . 95
5.7 Link break repair mechanism. . . . . 98
5.8 Average remaining energy in the network. . . . . 100
(a) Network size of 50 nodes. . . . . 100
(b) Network size of 100 nodes. . . . . 100
(c) Network size of 200 nodes. . . . . 100
5.9 Number of nodes with a specific energy consumption. . . . . 101
(a) T-MAC. . . . . 101
(b) N-CDS. . . . . 101
(c) MPR-based CDS. . . . . 101
5.10 Cumulative distribution function of energy consumption per node. 102
5.11 Average packet loss. . . . . 103
5.12 Area of additional coverage of neighbor nodes of $x$. . . . . 103
5.13 Backbone size evolution for R-CDS-LD/P and MCDS. . . . . 105
(a) Sparse - low mobility . . . . . 105
(b) Sparse - high mobility . . . . . 105
(c) Dense - low mobility . . . . . 105
(d) Dense - high mobility . . . . . 105
5.14 Average remaining battery level in the network for R-CDS-LD/P. . 106
(a) Sparse - low mobility . . . . . 106
(b) Sparse - high mobility . . . . . 106
(c) Dense - low mobility . . . . . 106
(d) Dense - high mobility . . . . . 106
5.15 Variation of energy consumption among the network nodes for R-CDS-LD/P. . . . . 107
(a) Sparse - low mobility . . . . . 107
(b) Sparse - high mobility . . . . . 107
(c) Dense - low mobility . . . . . 107
(d) Dense - high mobility . . . . . 107
5.16 Link break repair performance for R-CDS-LD/P. . . . . 108
(a) Low mobility . . . . . 108
(b) High mobility . . . . . 108
(c) Low mobility . . . . . 108
(d) High mobility . . . . . 108
5.17 Network topology and MCDS of the experiment. . . . . 110
5.18 Number of dominators in the experiments. . . . . 110
5.19 Link repair experiment. Node 504 is manually turned off . . . . . 111

5.20 Link break repair time in the different CDS cycles. . . . . 111

6.1 Event detection, tracking and reporting with DELTA. . . . . 116
6.2 State diagram of the node roles of DELTA. . . . . 118
6.3 Group communication in a DELTA tracking group. . . . . 120
6.4 Fraction of received messages for varying transmission power. . . 121
(a) Distance 1.25 meters . . . . . 121
(b) Distance 2.5 meters . . . . . 121
6.5 Tracking of small-area events. . . . . 123
6.6 Tracking of large-area events. . . . . 123
6.7 Nodes elected as leader in a specific simulation run. . . . . 124
(a) EnviroTrack. . . . . 124
(b) DELTA. . . . . 124
6.8 Experiment setup with 25 sensor nodes. . . . . 125
6.9 Event path through the sensor network. . . . . 125
6.10 Fraction of concurrent leaders in DELTA and EnviroTrack. . . . . 126
(a) Tracking of a 25W bulb . . . . . 126
(b) Tracking of a 40W bulb . . . . . 126
6.11 Communication costs of DELTA and EnviroTrack. . . . . 127

7.1 Localization and Signal Strength Estimation of SD, CG and LLS. . 131
(a) Distance Error . . . . . 131
(b) Signal Strength Error . . . . . 131
7.2 Localization accuracy of SD and LLS. . . . . 132
7.3 Accuracy of LLS, SD, and CG in an over-determined system. . . . 133
(a) Distance Error . . . . . 133
(b) Signal Strength Error . . . . . 133
7.4 Localization accuracy of SD and LLS in an over-determined system. 134
7.5 Arrangement of sensors **o** and event locations **x**. . . . . 135
7.6 Experiment setup with 15 event positions. . . . . 136

8.1 Mapping a cluster to the fuzzy sets $\tilde{\mathbf{A}}_{\mathbf{k,i}}$ of the premises in $\Re^{\mathbf{2}}$. . . 143
8.2 Light pattern I (upper part) and light pattern II (lower part). . . . . 152
8.3 Sequence of changes of classification outputs. . . . . 152
8.4 Occurrence of state changes in experiments 1 to 3. . . . . 153
8.5 Occurrence of state changes in experiments 4 to 6. . . . . 154

9.1 Office monitoring deployment. . . . . 159
9.2 System designs with local memory (left) or local threshold (right). 162
(a) Local Fuzzy ART. . . . . 162
(b) Local thresholds. . . . . 162
9.3 Anomaly Detection Performance with ART neural networks. . . . 167
(a) Soft hourly office searching. . . . . 167
(b) Intense hourly office searching. . . . . 167

# List of Tables


4.1 Parameters of the MAC simulations. . . . . 80
4.2 Percentage of coexisting schedules (taken from [85]). . . . . 83
4.3 Power consumption (in mAs) per listen/sleep cycle. . . . . 84

5.1 Backbone construction on MAC and networking layer. . . . . 86
5.2 Parameters used in the R-CDS simulations. . . . . 104
5.3 Duration / periodicity of the different relevant intervals. . . . . 109

7.1 Distance errors $\varnothing$ and standard deviations $\sigma$ on ESB nodes in [cm]. 135
7.2 Distance errors $\varnothing$ and standard deviations $\sigma$ on TmoteSky nodes in [cm]. . . . . 137
7.3 Signal strength errors $\Theta$ and standard deviations $\sigma$ on TmoteSky nodes in Lux. . . . . 137

8.1 Error rates of the classifiers. . . . . 150
8.2 Reporting delays of triangular FLC. . . . . 150
8.3 Reporting delays of FFNN. . . . . 151

9.1 False Positives (FP) and False Negatives (FN) of the anomaly detectors. . . . . 165
9.2 Message load of the anomaly detectors. . . . . 166
9.3 Reporting Latencies [s] until alarm is reported. . . . . 166

# Preface

The following work has been performed during my four-years employment as research and lecture assistant at the Institute of Computer Science and Applied Mathematics (IAM) of the University of Bern. The research conducted in this thesis has been supported by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation.

First, I would like to express my gratitude to my advisor, Prof. Dr. Torsten Braun, head of the Computer Network and Distributed Systems group (RVS), who supervised and encouraged my work. Prof. Braun gave me furthermore the opportunity to publish and present the work at multiple conferences, which involved helpful feedback that further improved the work.

I would also like to thank Prof. Dr. Peter Kropf who agreed to conduct the proofreading of this work. Prof. Dr. Oscar Nierstrasz who was willing to be the co-examinator of this work deserves many thanks too.

Many thanks go to my colleagues at the institute and in our research group, who have provided a friendly and pleasant working environment throughout my term of deployment. In particular, I would like to thank Markus Anwander, Marc Brogle, Thomas Bernoulli, Philipp Hurni, Dragan Milic, Matthias Scheidegger, Thomas Staub, Gerald Wagenknecht, Attila Weyland, and Markus Wulff. Special thanks go to Thomas Bernoulli, Thomas Staub, and Attila Weyland who shared their offices with me and gave me extra feedback and room for discussion. I am grateful to Dragan Milic for his helpful advices in the application of numerical optimization methods as well as for his regular running support.

I also would like to thank all the students who worked with me and helped a lot in developing and implementing the conducted work. Special thanks go to Markus Anwander, Samuel Bissig, Michael Meer, and Reto Zurbuchen, who performed their master's thesis with me. Special thanks go to Ruth Bestgen, the secretary of our research group, for her help and friendship during all these years. Finally, many thanks go to Peppo Brambilla, our system administrator, for making things a lot easier and for his friendship and running support.

I am grateful to my family, who supported me in many ways: My parents Heidi and Hans-Rudolf Wälchli and my sister Jolanda Reuteler with her family. Furthermore, I would like to thank my friends for support and motivation: Oliver Burkert, Kai Brünnler, Yasemin Gigandet and Tobias Roth are emphasized amongst them.

# Summary

In this thesis the development and implementation of an energy-efficient, fault-tolerant and fully distributed event detection system is presented. Long-term deployments must be supported, while satisfying the detection accuracy requirements of the application. On the one hand energy-efficient mechanisms are required to support long-term operation. While on the other hand, accurate event detection and classification methods are needed to satisfy the Quality of Service (QoS) requirements of the application. Current event detection systems are barely able to optimize the trade-off between QoS and long-term operation. Many existing event detection systems are designed for short-term deployments (personal security, disaster management). In such systems energy savings are less important. The deployment of a persistent surveillance system is desirable. Such a system is not useful if sensor node batteries have to be replaced frequently.

To provide a useful accurate event detection system for long-term surveillance, an integration of application-specific requirements is needed on different layers in the network stack. In our work we have proposed an architecture that implements a fully functional network stack designed to meet these application goals.

Medium access has been optimized to support long-term deployments. The synchronization messages used by synchronized contention-based MAC protocols have been exploited to setup a routing backbone. Non-backbone nodes turn their radios temporarily off to save additional energy. The mechanism requires no specific control traffic for routing. A simple clock synchronization scheme has been developed to decrease the duty cycle of backbone nodes. Upon detection of an event, all nodes become active and provide networking functionality to the application software. Thus, medium access and routing are completely transparent to the application layer. A mobility support module has been provided. The developed medium access and routing mechanisms have provided good energy consumption distributions in the network. The lifetime of nodes could be extended.

On top of the medium access and networking services the application-specific functionality has been implemented. Event tracking groups are dynamically established and maintained upon detection of an event. The tracking group organization is lightweight and effective. Arbitrary events are detected and tracked with nearly optimal signaling load. The tracking group further provides the event localization and classification software running at the group leader node with the needed information to perform these tasks. Hence, event localization, classification and filter-

ing are performed in the network. Thus, considerable reporting load can be saved, discharging sensor node along the paths to the base station. The distributed localization and classification features have shown efficient and accurate performance. The event detection system has been used in an building intrusion application that reports illegal office occupancy. The office monitoring system requires moderate communication and computation, hides the identity of office staff, and satisfies detection requirements. The system has proven to filter normal office occupancy in the network. Communication costs of approximately 90% could be saved. Every office intrusion has been detected and reported by the system. The prevention of false alarms has shown good performance.

Chapter 1

# Introduction

Wireless sensor networks are composed of large numbers of densely deployed sensor nodes. These sensor nodes are located in the proximity of the environmental phenomenon of interest. Due to limited sensing capability of the implemented sensors, which is caused by miniaturization, low-power operation, and low cost production, accuracy is achieved by massive parallelism. Error-prone sensor nodes and their inaccessibility due to remote installation require additional redundancy in order to provide reliable, unattended long-term operation. According to these characteristics, wireless sensor networks offer the monitoring of a physical phenomenon of interest from proximity with large numbers of cheap and tiny sensor nodes. On the other hand, the same characteristics require self-organization, coordination and distributed operation. Moreover, if the sensor nodes are battery-powered, which is commonly the case, energy-efficiency is a major criteria too.

The subject of this thesis is the development of a distributed event monitoring system for the detection and processing of events occurring unexpectedly in wireless sensor networks. We are in particular interested in the trade-off between detection accuracy and energy savings. The wireless sensor networks that we deploy are battery powered and intended for long-term operation. Accordingly, energy savings are a key issue in system design. On the other hand, event detection and classification are useless if accuracy requirements cannot be satisfied. In this thesis the trade-off between detection and accuracy is investigated in event detection applications of unattended long-term monitoring. An efficient event monitoring system requires distributed and tailored protocols on different layers of the network stack. At the application-layer, event detection procedures that satisfy given accuracy requirements, while saving as much energy as tolerable are needed. Network lifetime can further be extended by providing the event detection system with energy-efficient routing and medium access.

In event detection and monitoring systems, sensor networks are used to detect, track and classify suddenly evolving, often unexpected, events. Event monitoring systems can be used by many applications, ranging from sniper detection over building surveillance to object tracking and early warning systems. All these tasks require several system features in the application layer. Accordingly, specific event

detection, tracking and classification methods are provided in this thesis. There is a fast and efficient event detection and tracking algorithm, an event localization and signal strength estimation method, and two classifiers for the classification of discrete events and the detection of abnormal behavior, respectively. In order to minimize communication costs these algorithms should be performed distributed rather than static.

Finally, the deployment of the event monitoring system in an office access monitoring application is provided. Current building surveillance systems mainly deploy video surveillance systems. These kinds of systems are expensive and unpleasant to the working staff. On the other hand, building intrusion by thieves must be monitored in many systems. Therefore, we have investigated the development of a low price building monitoring system that hides the identity of the office staff.

In addition to the application-specific tasks, a set of methods that provide and optimize energy-efficient medium access and routing have been proposed. Thus, the lifetime of the monitoring system can be further extended. A simple clock synchronization scheme that minimizes overhead and energy consumption in synchronized contention-based MAC protocols is provided. Furthermore, synchronization messages exchanged on the MAC layer are used to setup a routing backbone on the MAC layer. In order to support node mobility an additional backbone mechanism has been implemented on the routing layer. All backbone construction mechanisms shut the radios of non-backbone nodes down temporarily to save additional energy.

In the remainder of this chapter, we first introduce the problems that are addressed in this thesis. Then, the general system architecture is presented. The remaining sections correspond to the main chapters, which provides more details about the research carried out for this thesis. The work is presented in ascending order according to the network stack beginning at the MAC layer. The last two sections contain a short summary of the main contributions of our work and give a brief overview over the structure of the thesis.

## 1.1 Problem Statement

Even though event detection systems have been researched in numerous projects, there is no system that provides lightweight, accurate, and persistent building monitoring. Current systems are very application-specific and can hardly be adapted to different monitoring tasks. Long-term deployment is generally not in the scope of existing systems. On the other hand, systems that might support long-term deployments often lack accuracy. The detection of suspicious (abnormal) behavior has barely been addressed yet. However, anomaly detection simplifies the classification problem and can contribute to a lightweight surveillance system. With our system we aim at developing a lightweight building surveillance system.

Current solutions either typically address mainly subtopics or provide very specific solutions that are hardly applicable to more general system designs. We classify current event detection systems into the following three categories:

- **Non-permanent systems**: The main characteristic of such a system is the subordination of long-term deployment, because accurate information is only temporarily required. A typical application is a sniper detection system (see Section 3.5.6). Such a system is temporarily installed. Both, energy savings and communication minimization are relinquished in order to achieve very accurate localization, e.g., of a sniper. Finally, the sensor network is deployed according to detailed installation plans and the sensor node redundancy is low. Such systems are not applicable to more general sensor network systems that aim at long-term operation.

- **Accurate systems**: The main goals of many event detection systems, which are often also addressed by non-permanent systems, are localization and classification accuracy optimization. Research mainly focuses on collaborative signal processing (CSP). The approaches require high communication and computation load. Networking issues such as event tracking, efficient medium access control, topology control, and communication optimization are hardly considered. Typical representatives of such systems have been developed in the SensIt project (see Sections 3.5.8 and 3.6.7).

- **Efficient systems**: On the other hand, there are event detection and tracking algorithms that barely address localization and classification tasks. Such systems are able to detect and track an occurring event fast and efficiently. They do not provide sufficient sensor data to make reasonable statements about the observed event, though. Such systems are dedicated if the observed event or phenomena is well known. However, if the sensor system itself should make conclusions about the kind of observed event, such systems need to be enhanced with other features. A typical system of this kind is EnviroTrack, which is discussed in Section 3.4.5.

The current state of the art work is either tailored to very specific applications requiring short-term deployment, focusing mainly on the optimization of classification accuracy, or optimizes event detection and tracking. In this thesis we aim at providing an energy-efficient system that provides the required event detection and tracking accuracy in long-term deployments. The requirements on our system can be defined as follows:

- **Detection accuracy**: Our system must be able to satisfy event detection requirements of the application. In our building monitoring system this means that building intrusion must be signaled, while false alarms should be prevented.

- **Long-term deployment**: Our system is intended for persistent monitoring. Hence, the algorithms should be lightweight. Whenever possible, energy should be saved by the system.

Each of the algorithms proposed in this thesis aims at achieving these goals. The solutions on the MAC and routing layer provide energy-efficient medium access control and routing to extend network lifetime. The application layer divides event detection and tracking into three subproblems. First, the detection of events and the management of tracking groups is addressed by a dedicated protocol. This protocol also covers the collection of event-relevant data by the tracking group leader. Second, the collected data is processed at the leader node to estimate event characteristics such as emitted signal strength(s) of the event or its location. Third, events are classified at the leader node, optionally based on the event characteristics computed in the previous step or on raw data that has been collected. The resulting system has been applied to office monitoring.

## 1.2 System Architecture

In the following the architecture of our event detection system and a typical deployment scenario are illustrated. The network stack of the general system architecture is shown in Fig 1.1.

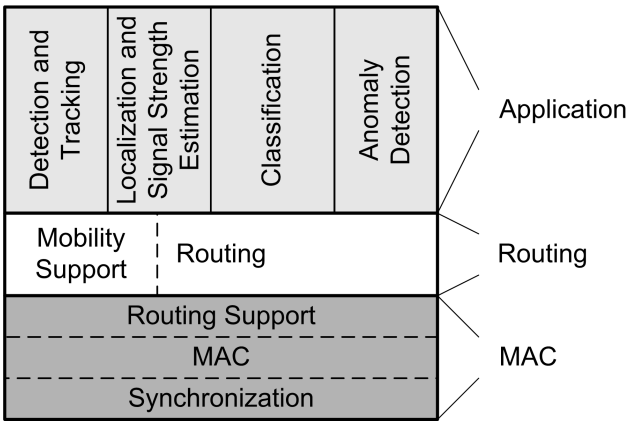



**Figure 1.1:** General system architecture.

Synchronization, medium access and routing support functionality is implemented on the MAC layer, i.e., on the lower layers in Figure 1.1. This layer focuses on providing the event detection system with energy-efficient medium access and routing support. Above the MAC layer is the networking layer. This layer provides simple routing based on the backbone provided by the MAC. In addition, the mobility support module can be used if the network topology changes frequently (e.g., in mobile networks). Finally, all functionality relevant for event detection is implemented on the application layer, which is shown in Figure 1.1. This layer contains the detection and tracking algorithm, the localization and signal strength estimation procedures and the classification and anomaly detection modules. Every sensor node implements a network stack containing the described functionality and

modules. The system components are introduced in the next sections.

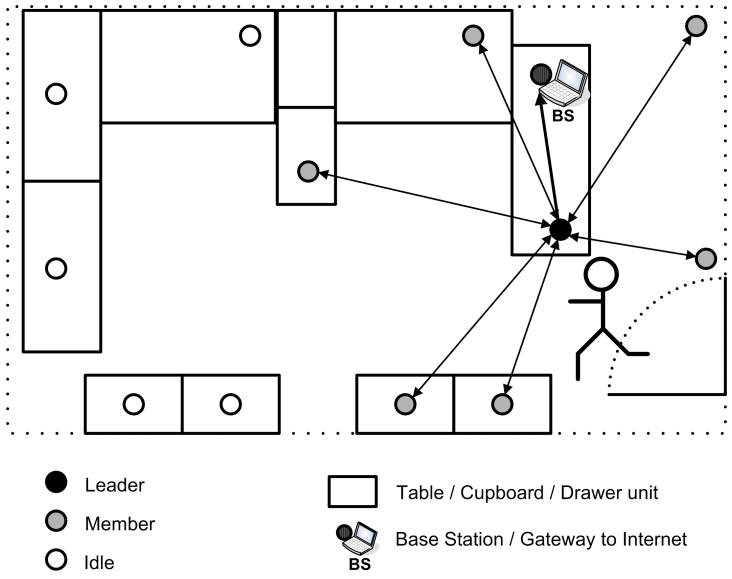



**Figure 1.2:** Typical deployment scenario.

A typical application of our event monitoring system is illustrated in Figure 1.2. A simple office monitoring example is depicted. The office is monitored by a number of sensor nodes (the circles in Figure 1.2). Upon occurrence of an event, e.g., a person enters the room, the surrounding sensors form a tracking group, collecting and processing local sensor measurements. Event reports (containing position or classification results) are computed at the group leader node and are forwarded to the base station, which has access to the Internet. If the person was moving to the workplace on the left, the tracking group would be reorganized. The idle nodes on the left in Fig 1.2 would participate in the newly established tracking group.

## 1.3 System Evaluation

The event detection system has been evaluated both in simulations and in real-world experiments. All simulations throughout the whole thesis have been implemented and performed in the OMNeT++ [144] network simulator. For the real-world tests two common sensor network platforms have been considered. Some parts have been implemented and tested only on the Embedded Sensor Boards (ESB) platform (see Section 3.8.1). Some functionality has been re-implemented on TmoteSky sensor nodes (see Section 3.8.2). The relevant difference between both platforms for our experiments is the implemented radio. Therefore, some functionality has been re-implemented on the TmoteSky platform to investigate performance with more powerful and more reliable radio communication.

## 1.4 Medium Access and Routing

Chapters 4 and 5 present the functionality on the lower two layers of our architecture (see Figure 1.1). The goal of our work in medium access control has been to provide our event monitoring system with energy-efficient medium access and topology control. The wireless sensor network we envision deploys large numbers of typically static sensor nodes. To provide essential medium access, synchronized contention-based MAC protocols have been used. These protocols implement low duty cycles. To maintain connectivity, the duty cycles need to be synchronized. Therefore, synchronization messages are periodically exchanged among the sensor nodes.

The synchronization of duty cycles maintained in wireless sensor networks is addressed in Chapter 4. We introduce a simple algorithm which converges towards the usage of a common duty cycle for all wireless sensor nodes. The algorithm does not need any additional control traffic, but solely exploits relations in the number of transmitted synchronization messages. Moreover, we have exploited the information intrinsically provided by the synchronization mechanism to conserve additional energy on the MAC layer as well as to provide routing in Chapter 5. We have used the neighborhood information, which is intrinsically provided by the synchronization messages, to implement two routing backbone mechanisms that are based on connected dominating sets (CDS) directly on the MAC layer. Thus, no specific routing control traffic is generated. Implementing a routing backbone on the MAC layer introduces some setup delays and prevents local path adaptations and repair mechanisms. Therefore, no node mobility is supported by these mechanisms. In order to compensate for this drawback, an additional CDS-based backbone mechanism has been implemented on the routing layer. The algorithm requires extra traffic, but supports more dynamic networks. All backbone construction mechanisms turn non-backbone nodes temporarily off to save energy. The main outcomes of this part can be briefly summarized as follows:

- A common duty-cycle of nodes running synchronized contention-based MAC protocols is achieved by exploiting the number of synchronization messages sent by clusters of sensor nodes.
- The content of the synchronization messages is used to implement routing directly on the MAC layer. Thus, no additional control traffic is required.
- To account for more dynamic network topologies, an additional backbone construction mechanism is proposed on the network layer. This mechanism requires specific control messages, though.

## 1.5 Event Detection and Tracking

Chapter 6 addresses the detection and tracking of moving objects, i.e., functionality of the first application module in Figure 1.1. A fast and energy-efficient

distributed event localization and tracking algorithm (DELTA) is proposed. The algorithm detects and tracks moving objects according to local sensor readings. A distributed leader election and group maintenance mechanism is provided. The distributed communication and management scheme avoids heavy data traffic towards the base station, which disburdens nodes close to the base station and prevents early network fragmentation due to drain of central sensor nodes. In addition to tracking group formation and maintenance, the leader node is further responsible for leader handover, data collection and processing, and event reporting. The basic functionality, i.e., event detection, tracking group formation, data collection and processing, and event reporting are illustrated in Figure 1.2. Upon appearance of an event (the person has entered the room), the sensor node with the highest sensor readings is elected as leader (the black node in Figure 1.2). The leader immediately starts requesting sensor readings from its neighbors (the group members in Figure 1.2). With this mechanism the group members are informed about the leader and further provide the leader with event-relevant data needed for localization and classification tasks.

Tracking is performed by handing over the group leader state. Arbitrary sensing ranges can be supported due to an optimized state dissemination procedure. Two-hop neighbors of the leader are intrinsically informed about the tracking group by overhearing the response messages of the group members. The tracking group state can optionally be distributed deeper into the network by an optimized broadcasting technique. Thus, the appearance of concurrently present tracking groups can be prevented. This is important because the existence of multiple tracking groups implies communication overhead. Multiple tracking groups would result in multiple event reports, which would excessively charge nodes close to the base station by having to forward all those reports. On the other hand, the communication costs of the state dissemination process also increase with higher sensing ranges. However, this cannot be avoided if the appearance of concurrent tracking groups has to be prevented. The trade-off can be estimated, though. Finally, communication costs are well distributed with DELTA.

DELTA has been designed to run on tiny sensor nodes. Simulation and real-world experiment results are presented. The evaluation shows that the algorithm works efficiently, minimizing the amount of communication, while providing sufficient information to make meaningful statements about location and type of the observed event. DELTA provides the following main features:

- Efficient formation and maintenance of tracking groups based on sensor readings.
- Concurrently present tracking groups are avoided.
- The required sensor data to perform fine-grained localization and classification is collected in a distributed manner.
- Arbitrary sensing ranges are supported.

- Communication load is minimized.

## 1.6 Event Localization and Signal Strength Estimation

The localization and signal strength estimation module (see Figure 1.1) is introduced in Chapter 7. The DELTA algorithm proposed in this thesis provides the leader node with event-relevant sensor readings. This data is required for the computation of event characteristics such as the signal strength(s) emitted by an event or its location. Furthermore, any subsequent classification can be based on these estimated event characteristics. Only the computed estimates are reported to the base station. Thus, considerable communication load can be saved.

The computation of event position and emitted signal strength(s) of the event is based on local sensor readings. In order to compute these estimations, a feasible signal propagation model is required. In our work we use a commonly applied sensor model, which assumes isotropic signal attenuation. The received signal strength on a sensor node decreases thereby inversely proportional to the distance to the event source. The sensor model is used to formulate the localization and signal strength estimation problems as a nonlinear objective function. In order to solve this function, at a minimum $n$ sensor readings are needed, whereby $n$ is bigger than the problem dimensionality. The collection of data is illustrated by the information exchange between the group leader and the group members in Figure 1.2. Thereby, the group leader collects the data required to perform the localization and signal strength computation tasks. The objective function is solved by applying nonlinear optimization methods. The problem can further be linearized and solved with linear least square methods. In this case, additional sensor readings are required. The availability of redundant information cannot always be guaranteed in wireless sensor networks, though.

Both, simulated events as well as real-world events have been evaluated. The results show accurate localization and event parameter estimations. We have evaluated nonlinear and linearized solutions. Nonlinear solutions have shown to be superior. In particular, they provide useful estimations even when only a minimum amount of data is available, which might frequently be the case in wireless sensor networks due to error-proneness or energy (communication cost) constraints. The main findings of this part can be summarized as follows:

- Localization and parameter estimation problems are formulated as nonlinear optimization problem based on collected sensor readings.
- Nonlinear solutions outperform the according linearized methods.
- Reporting traffic is minimized.
- The classification of events based on event characteristic estimates is facilitated.

## 1.7 Event Classification

Finally, event classification methods have been investigated in Chapter 8. Two different classification methods have been developed. The first method addresses the classification of discrete events and is presented in Section 8.2, while the second method concerns the detection and reporting of abnormal behavior. The first method aims to classify events which are present as discrete entities in time (Module 3 in Figure 1.1). With this method a certain number of well-defined time-discrete event types can be distinguished. The time-discrete event classification procedure learns and classifies specific event patterns unsupervised from collected data. Learning from data has the advantage that no expert knowledge is required. Thus, the application of the algorithm is simplified and design flaws due to poor data abstractions can be prevented. Moreover, the system supports false-alarm filtering, which can save costs in terms of energy and money. The classifier assigns a confidence degree to each classification and filters events that do not satisfy a given threshold requirement. Obviously, filtering has an impact on reporting delays. The trade-off between false-alarm prevention and introduced reporting latency is investigated. Regular alarms, commonly detected with high confidence, must not be filtered, but reported with low latency. The main outcomes of this part can be briefly summarized as follows:

- Discrete event classes are learned and classified in an unsupervised manner.
- Classifications are rated with a certain confidence. Based on this confidence false alarms are prevented.
- Regular alarms are reported with short delays.
- The classifier is configured offline based on collected data. The classification itself is lightweight and is performed on the leader node based on current sensor data.

## 1.8 Anomaly Detection

The classifier for discrete events cannot be extended to model events which evolve over time. To address these kinds of events, which are often present in surveillance and tracking applications, dedicated lightweight methods are required. The classification of continuously evolving events is very expensive in terms of communication and storage. On the other hand, our event detection system mainly requires anomaly detection. This poses less burden on the system. Surveillance applications are more interested in detecting abnormal behavior than determining specific event characteristics. The method is proposed in Section 8.3. Anomaly detection is covered by the fourth application module in our architecture (see Figure 1.1). The approach implements an adaptive memory on the sensor nodes that is able to remember event patterns. Abnormal behavior is identified by unknown

event patterns. The adaptive memory approach is implemented as short term memory based on an aging mechanism. Thereby, stored event patterns that rarely occur become older. Knowledge that exceeds a given age is replaced by new, currently unknown event patterns. Based on a rather small memory capacity and the aging mechanism, currently unknown events are detected, reported as anomaly, temporarily known and replaced by new, unknown event patterns. The main findings of this part can be summarized as follows:

- Abnormal behavior is detected and reported with an adaptive memory approach.
- Known event patterns are filtered on the nodes. Communication is saved.
- Based on the aging mechanism, learning capability is continuously provided.
- Anomaly detection can be implemented in a lightweight and efficient manner by adding aging functionality.

## 1.9 Office Monitoring Application

The event detection system has been used in an office monitoring application, where unauthorized office occupancy is detected and reported. The application of our event detection system is presented in Chapter 9. The anomaly detection software and the DELTA event detection and tracking functionality are used. A typical monitoring scenario is illustrated in Figure 1.2. Different office occupancy or office access patterns need to be distinguished. The access patterns are composed of a series of measurements of some phenomena that are collected and processed on the sensor nodes. In order to meet storage requirements the office monitoring has been restricted to anomaly detection, i.e., no classification of present event patterns is performed. Anomaly detection conceals the identity of persons occupying the office in a normal state, because only abnormal behavior is reported. Thus, the system provides privacy to the office staff. Due to energy costs of radio transmissions, it is not possible to transmit the observed event patterns unprocessed to a fusion center. Therefore, the office monitoring system implements a two-layered approach. Local event patterns are periodically monitored, filtered and compressed on the sensor nodes. The compressed event report is then sent to a fusion center, i.e., a DELTA leader node, where the system-wide anomaly detection is performed. The main outcomes of Chapter 9 can be briefly summarized as follows:

- Unauthorized office access is reliably detected. Normal office occupancy triggers no alarms.
- Unauthorized office access is reported efficiently and in a lightweight fashion.
- The office monitoring system provides privacy to the office staff.

## 1.10 Contributions

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

- A simple synchronization method for synchronized contention-based MAC protocols has been developed. The method prevents coexisting duty cycles and saves additional energy on the MAC layer.
- Synchronization messages have been used to implement routing on the MAC layer. Thus, additional control traffic is saved. Additional energy is saved by temporarily turning off the radios of nodes that are not required for routing.
- Network lifetime has been extended by tailored medium access and routing.
- Tracking groups are efficiently established and maintained. DELTA outperforms similar approaches in communication overhead minimization.
- An energy efficient event detection architecture that supports long-term deployments has been developed. The system provides long-term monitoring without need for battery replacements, which has not yet been provided by other event detection systems.
- The trade-off between long-term operation and detection accuracy has been optimized, while previous work mainly focuses on accuracy optimization.
- Discrete event classes are efficiently and unsupervised classified.
- Lightweight and accurate anomaly detection is performed by an adaptive memory approach.
- The event detection system has been applied to office monitoring. The system works efficiently and reliably.
- Office intrusion by thieves is detected and reported reliably. Normal office occupancy is filtered in the network, decreasing communication costs. Privacy is provided to office staff.

## 1.11 Thesis Outline

In Chapter 2 an introduction to wireless sensor networks is given. The focus is on relevant characteristics for this thesis. The network stack layers of interest are introduced and discussed in some more detail.

In Chapter 3 we give a comprehensive overview of related work in the topics of medium access control, event detection, tracking group formation and maintenance, event localization, and event classification for wireless sensor networks.

A simple protocol to achieve local clock synchronization is presented in Chapter 4. The method is used to synchronize duty cycles of sensor nodes running synchronized contention-based MAC protocols.

Energy efficient protocols that provide our event detection system with medium access and routing are presented in Chapter 5. The protocols exploit the information exchanged by synchronization messages. Based on this information routing backbones are implemented directly on the MAC layer. To save additional energy, non-backbone nodes temporarily turn off their radios.

The remainder of the thesis discusses the different application-level tasks. Chapter 6 addresses event detection and the formation and management of tracking groups. These operations are performed by the DELTA algorithm. DELTA is evaluated in terms of tracking performance and communication overhead minimization.

A method to estimate location and signal strength(s) of events is presented in Chapter 7. The method is based on nonlinear function optimization and is implemented as a feature of DELTA.

Event classification procedures are provided in Chapter 8. The procedures contain a classifier based on Fuzzy Logic concepts that learns event classes unsupervised and classifies unknown events quickly and efficiently. Furthermore, a classifier that performs anomaly detection by monitoring continuously present event patterns is provided.

The event detection system has been deployed in an office monitoring application. Deployment and performance details are presented in Chapter 9.

Chapter 10 summarizes the main outcomes of this thesis and concludes the performed work.

Finally, some further improvements and possible future directions of research are presented in Chapter 11.

Chapter 2

# Wireless Sensor Networks

In this chapter a general introduction to wireless sensor networks (WSN) is given. Focus is on topics that are relevant for this work. Wireless sensor networks are sometimes considered as a special kind of mobile ad-hoc networks (MANET) [109]. However, the two networks differ considerably. Wireless sensor networks are deployed to monitor and interact with physical environment, whereas MANETs have been developed to interconnect mobile computers in an ad-hoc and infrastructureless manner. Moreover, miniaturization, long-term deployment, difficulty in physical access and high redundancy impose very different requirements to wireless sensor networks compared to MANET.

## 2.1 Introduction

Recent advances in wireless communication technology and the development of low cost, low power, multifunctional sensor nodes have led to the development of wireless sensor networks. The tiny sensor nodes consist at a minimum of a processing unit, some memory, a radio module to exchange data and an array of sensors to measure physical phenomena. In addition, sensor nodes are generally equipped with batteries. Consequently, the ability to save energy in order to extend node lifetime is a critical evaluation factor in most applications. Sensor nodes can optionally be provided with actuators to interact with the physical environment.

Wireless sensor networks are a significant improvement from traditional (normally wired) sensor systems, which provide solutions to problems in the following two contexts [3]:

- Large powerful sensors are positioned far from the actual physical process of interest. In this approach few heavily-equipped high-resolution sensors are deployed, which provide complex techniques to measure and filter physical phenomena.

- The physical phenomenon is observed by several sensors that perform only sensing and transmit the observed raw time series of measurements to a central fusion center, where the streams of sensor readings from the different

sensors are stored and processed. In-network data processing is not supported. The positions of the sensors and the network topology need to be engineered carefully. Moreover, if the sensor needs to be wired, high deployment costs are a consequence.

In contrast wireless sensor networks are composed of a high number of densely deployed nodes that are located close to the monitored physical phenomenon. The sensor nodes are often assumed to be randomly deployed in terrains that are difficult to access, e.g., in remote areas or in disaster areas. This introduces flexibility, but this also imposes complexity. Wireless sensor networks have to provide self-organizing capabilities. Moreover, remote access and reconfiguration functionality as well as redundancy to compensate for node failures are required.

In order to fulfill an application-level task, sensor nodes are commonly required to operate cooperatively. Because sensor nodes are equipped with a processing unit and some storage, raw sensor data can be processed in-network, on node-level or iteratively at dedicated sensor nodes. These dedicated nodes can be determined based on negotiation procedures, by other simple election methods, or by topology control mechanisms. Thus, the transmission of huge amounts of raw sensor data to a central fusion center can be avoided. This saves communication costs and accordingly energy.

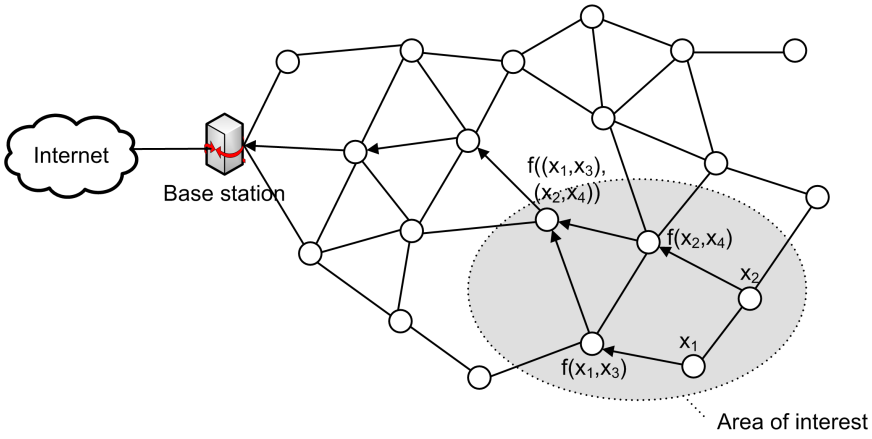



**Figure 2.1:** Wireless sensor network with in-network processing.

A typical example of a wireless sensor network is depicted in Figure 2.1. The network is connected to the Internet over a base station. A high number of sensor nodes are connected and build a multi-hop wireless network. In a given area of interest sensor data is collected, aggregated and finally routed to the base station for further processing. In Figure 2.1, the five sensor nodes located in the area of interest are organized in a tree structure. Thus, by applying aggregation along the tree, only one report, containing the aggregated data from all five sensor nodes, is sent to the base station. This sensor network example requires collaboration, routing support and aggregation functionality.

Many researchers are engaged in developing algorithms that address these requirements. More recently, the focus has changed from more theoretical work and simulations towards real-world implementations and prototypes. Good introductions to wireless sensor networks, some of its specific topics, and its applications have been provided by [3], [2], [34], [35] and [112]. Before presenting the work done in this thesis, some concepts and properties of wireless sensor networks that are in particular relevant for this thesis are discussed in more detail. The spectrum of sensors implemented on sensor nodes is listed in Section 2.2. Impacts of energy constraints and other influencing factors on sensor networks and their applications are discussed in Section 2.3. Section 2.4 addresses communication in wireless sensor networks. Focus is on medium access, routing and on the application layer.

## 2.2 Sensing Capabilities and Sensors

In the following the spectrum of sensors typically implemented on sensor nodes is presented. Sensor networks can consist of many different types of sensors such as acoustic, light, thermal, accelerometer, infrared, seismic and visual. The implementation and combination of these kinds of sensors supports the monitoring of a wide variety of ambient conditions that include, but are not limited to, the following [36]:

- Temperature,
- Humidity,
- Movement and velocity,
- Light condition,
- Pressure,
- Soil conditions,
- Noise levels,
- The presence or absence of certain kinds of objects,
- Mechanical stress levels on attached objects, and
- Temporary characteristics such as speed, direction, and size of an object.

Due to harsh constraints in power supply and usage, the sensor nodes and the sensors implemented on them are comparatively cheap, provide limited accuracy and mainly support proximity sensing. On the other hand, dense deployment and massive parallelism, which are offered by the cheap cost of wireless technology, balance these drawbacks. The sensor nodes can be used for event detection, continuous monitoring, event ID handling, localization and classification, and the control

of actuators to feedback to the environment. The introduced concept of parallel micro-sensing together with wireless communications makes many new application areas accessible (see Section 2.4.5).

## 2.3 Factors Influencing Sensor Network Design

The design of a sensor network has to consider many factors. Typical system aspects are hardware design, low power operation, manufacture costs and the transmission medium. System aspects are discussed in Section 2.3.1. In addition to system aspects networking requirements influence the design of sensor networks. Typical networking issues are sensor network topology, scalability and fault tolerance. The factors are discussed in Section 2.3.2. Finally, environmental factors are discussed in Section 2.3.3. In order to develop practical sensor network solutions all factors have to be considered together. These factors provide a guideline for the design of hardware, protocols and algorithms for sensor networks. Moreover, they can be used for the comparison of different solutions.

### 2.3.1 System Issues

In the following the impact of sensor system aspects on the design and development of wireless sensor nodes and networks is discussed. These factors mutually influence each other.

#### Hardware Design

A sensor node consists of four basic components, namely a sensing unit, a processing unit with some storage, a transceiver unit and a power supply unit. Additional units such as a location service or a mobilizer might be implemented too. The analog signals captured by the sensing unit are typically converted to a digital output. All these units need to be fitted into a matchbox-sized module [56]. In addition to size, some other constraints have to be considered in the hardware design [3]:

- Energy consumption should be as low as possible.
- The nodes must operate at high volumetric densities.
- The components must be cheap and individual nodes must be dispensable.
- Autonomous and unattended operation is required.
- The nodes must be adaptive in respect to the environment.

Because sensor nodes are often inaccessible and/or deployed in very large numbers, the lifetime of a sensor network depends on the lifetime of the power supply unit. The developers of the ESB sensor nodes (see Section 3.8.1) have estimated a sensor node lifetime of 17 years if 25 bytes are sent every 20s [9]. The rest of

the time the sensor node is in very low power mode, in which the sensors, the radio and the processing unit are shut down. If synchronization, MAC and routing support, and other mechanisms are required, the lifetime drastically reduces, ranging from several months to a few years. The kind of transceiver unit used has a major impact on the sensor network design. In general radio frequency (RF) with bandwidths of up to few hundred kbits is used. Transmission range and reliability of the transceiver commonly vary considerably over time and are dependent on the node. Therefore, calibration might be required. Finally, the storage capacity of sensors nodes is limited. Implementing additional memory (e.g., EEPROM) is possible, but writing to that memory is expensive in terms of energy. Accordingly, sensor network applications and protocols are assumed to use as little memory as possible.

### Low Power Operation

Probably the most important constraint on sensor networks is the requirement of low power operation. Sensor nodes are in general equipped with low power batteries. Due to the paradigm of unattended operation, these batteries are often not replaceable, or only with high costs. Therefore, wireless sensor networks focus on the optimization of the trade-off between quality of service (QoS) and energy conservation. Considering routing, the trade-off is the option of prolonging network lifetime at the cost of increased delays and/or lower throughput. In localization and classification, the trade-off is normally between saving energy and increasing accuracy. Depending on the application QoS cannot be lowered in order to save energy, though.

Nevertheless, any application has to work within the power constraints of the used sensor nodes. Wireless sensor nodes can only be equipped with limited power supply. According to [3] currents of up to 0.5 Ah can be supported. Moreover, the sensor nodes are typically powered with up to three 1.5 V batteries, which results in a total power supply of 4.5 V. In specific applications the batteries can be recharged, whereas in most applications this is not the case. Therefore, the lifetime of a sensor network is strongly dependent on the lifetime of the supplied batteries. If the batteries of critical nodes deplete, network connectivity can be affected and routing can become impossible. Accordingly, the application might no longer be able to perform its tasks. To prevent this, the development of power-aware algorithms is one of the primary design factors for wireless sensor networks. Power consumption can be divided into three domains: sensing, communication, and data processing. The frequency of sensing is application-specific. Nevertheless, to reduce energy consumption the trade-off between sampling frequency and minimum required resolution is optimized. With regards to communication and data processing it has been shown that the communication of a bit in general costs much more than processing a bit [171]. Accordingly, energy can be conserved by processing and aggregating sensed data within the network.

#### Manufacture Costs

Wireless sensor nodes are comparatively cheap, but deployed in large quantities. In summary, the costs of a sensor network may not exceed the costs of deploying traditional sensors to address the problem at hand. In order to make the deployment of a large sensor network feasible, the costs of a single sensor node should be less than $1 [117].

#### Transmission Media

Communication in a multi-hop sensor network is achieved by wireless communication. The wireless medium can be radio, infrared or optical. In most applications radio is used. The medium should be available worldwide to support global operation. Often the license-free industrial, scientific and medical (ISM) bands are used. Due to constraints in power usage and size, the range of applicable carrier frequency is limited to the ultrahigh frequency range. Currently used transceivers mainly work in the 868 MHz or the 2.4 GHz spectrum [127], [128]. The used license-free frequencies are also often used by other applications. Accordingly, inter-application interferences will arise and have to be dealt with.

### 2.3.2 Networking Issues

The following aspects concern networking issues. The expected sensor network topology, the required network size and needs of fault-tolerant operation influence the design of wireless sensor networks.

#### Sensor Network Topology

The high number of densely deployed nodes in a sensor network makes topology maintenance a challenging task. The maintenance of topology can be mainly divided into two phases, namely the pre-deployment and deployment phase as well as the actual post-deployment maintenance phase. The deployment of sensor nodes can be done randomly by throwing them out of planes or by a catapult, or in a more controlled way by placing them in factory and/or manually by humans or robots in the target environment. The kind of deployment depends on the application. In any case, cost expenditure in planning and installation has to be considered. Even if the sensor network is statically deployed, the topology can change due to node failures, energy depletions, or jamming attacks after deployment. Moreover, varying battery levels among the network nodes might require rearrangements. In mobile scenarios or in the presence of node failures the need for topology maintenance is obvious. However, application-specific changes might require topology adaptations too. Finally, additional nodes might be deployed, which have to be integrated. The handling of topology changes is in general addressed on the network layer by dedicated routing and topology control protocols.

#### Scalability

Depending on the kind of phenomenon that has to be monitored and the paradigm of massive parallelism to compensate for rather local sensing, the number of deployed sensor nodes for studying the phenomenon is quickly on the order of hundreds and thousands. Depending on the application, the number might even reach an extreme value of millions [3]. In addition, the density has a large impact and can vary from a few nodes in a region up to thousands. The network density $m$ in a particular region $A$ can be computed according to [14]:

$$m = \frac{n \cdot r^2 \cdot \pi}{A}$$

where $n$ is the number of nodes deployed in a region and $r$ the radio transmission range. The number of nodes in a region is used to indicate node density. The computed network density is based on the unit disk model [24]. Hence, the estimated network density is very idealistic and does not consider irregular transmission ranges. Nevertheless, in order to get an approximation of the real network density, this simplistic model is commonly used.

#### Fault Tolerance

Sensor nodes are known to become completely or temporarily unavailable due to physical damage, lack of power or environmental interferences. The (temporary) failure of nodes should not affect the overall performance of the sensor network. In particular the satisfaction of the global task of the sensor network may not be corrupted. Accordingly, fault tolerance issues go hand in hand with system reliability. An introduction into the design of fault-tolerant sensor network systems has been provided by [53]. The deployment environment has a big influence on reliability. In a battlefield sensor nodes might be damaged much more frequently than in a building surveillance application.

### 2.3.3 Environment

The phenomenon of interest is observed by densely deployed sensors which are located in proximity to the monitored phenomenon. Since the phenomenon of interest is often located in remote or inaccessible geographic areas, the sensor nodes have to work unattended.

Harsh conditions that the sensor nodes may face include wild animals and other moving objects, harsh environments such as glaciers, hurricanes or oceans, interiors of machinery, biological or chemical contaminated fields, and so on. All these different kinds of environments pose challenges to the development of sensor nodes and networks.

## 2.4 Communication in Wireless Sensor Networks

In this section some communication issues in wireless sensor networks are discussed. In order to operate a WSN, different functionality on different layers of the network stack is required.

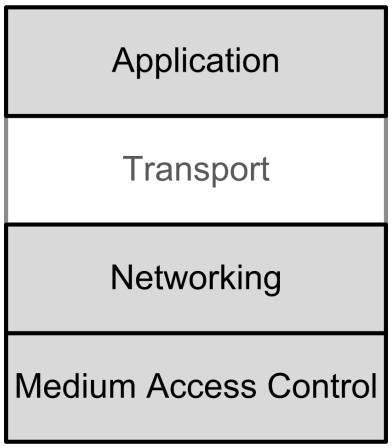



**Figure 2.2:** Network stack implemented by sensor nodes.

A typical network stack implemented on a sensor node is depicted in Figure 2.2. It provides similar layers as the stack known from the Internet. The layers that are of particular relevance for this thesis are colored gray in Figure 2.2. The transport layer provides functionality for reliable end-to-end communication. This issue is less considered in our work. Also, flow control and congestion control have not been in the scope of this thesis. Therefore, the according functionalities are not introduced in the following. Nevertheless, transport functionality and/or congestion and flow control could be integrated into our system.

Medium access control and routing are functionalities that every wireless sensor network has to provide. First, to distinguish Wireless Sensor Networks (WSN) from Mobile Ad Hoc Networks (MANET), we classify communication requirements of WSNs with respect to the communication requirements of MANETs. Then we introduce and motivate the usage of multi-hop communications in sensor networks. The focus is on aspects of medium access and routing, which are needed by our system. Finally, some typical application issues are discussed.

### 2.4.1 Requirements

The realization of sensor network applications requires wireless ad-hoc networking techniques (see also Figure 2.1). Although, many protocols have been presented in the context of MANETs, they do not meet the specific features of wireless sensor networks. In the following a list of the basic differences between WSNs and MANETs [109] is given according to [3]:

- The number of sensor nodes in a sensor network might be several orders of magnitude higher than the number of nodes in an ad hoc network.
- Sensor nodes are often densely deployed, which requires topology control mechanisms.
- Sensor nodes are error-prone, which requires redundancy.
- The topology of a sensor network can frequently change. However, this is less of an issue to the kind of event-detection systems addressed in this work.
- Sensor nodes are limited in power, computational capacity, and memory. Nonetheless, in most applications they must provide long-term operation.
- In some applications, sensor nodes are not addressed by global identifiers (IDs) due to the large number of sensors and overhead. Data-centric communication might be more adequate for such applications.

### 2.4.2 Multi-Hop Communication

Wireless sensor networks generally perform multi-hop communication. This is mainly for three reasons. First, large numbers of sensor nodes are densely deployed and neighboring nodes are accordingly located very close to each other. Traditional single-hop communication which would cover many nodes in such scenarios, would lead to unnecessary overhearing and therefore a substantial waste of energy. Hence, multi-hop communication is expected to consume less energy in sensor networks. Secondly, according to the dense deployment of sensor nodes, transmission power levels can be kept low, which saves a lot of energy. Finally, with multi-hop communication some signal propagation problems known from long-distance wireless communication can be overcome.

### 2.4.3 Medium Access

The objective of Medium Access Control (MAC) is to coordinate the times when a number of nodes access a shared communication medium. In addition to the common task of organizing the medium access by mutual exclusion mechanisms or assigning fixed transmission slots, MAC protocols for wireless sensor networks need to be energy-efficient. Basically, there are five sources of energy waste, which have to be considered in protocol design [30]:

- **Collisions**: Interference between concurrently sending nodes needs to be avoided, else the transmitted packets might be damaged. Moreover, the energy used for transmission and reception of these packets is wasted and the packets need to be retransmitted.
- **Overhearing**: Since air is a shared medium, nodes receive packets that are not destined for them. The energy used for receiving and processing these

packets is wasted. Nodes that do not take part in communication could preserve much more energy in a sleep state.

- **Overhead**: MAC protocols need control messages for protocol tasks such as synchronization, medium allocation, etc. These packets do not contain application data. Accordingly, these packets are overhead from the application point of view. The number of them should be minimized.

- **Idle listening**: The wireless medium is continuously sensed in order to detect pending transmissions from a node. Typically, idle listening consumes nearly as much energy as receiving.

- **Overemitting**: This is caused by the transmission of packets to a receiver when the receiver is not ready. This is obviously a waste of energy and requires retransmissions.

Apart form facing the sources of energy waste presented above, sensor networks need to shut down the radios of the nodes as often as possible to save extra energy. The goal is to minimize the duty cycle of the node, i.e., the amount of time the node is awake. The temporary shut down of radios can either be done in a synchronous or asynchronous manner. Low duty cycling challenges the satisfaction of traditional performance measurements such as delay and throughput. Some protocols minimize duty cycles, while trying to meet application-specific delay and throughput goals. A classification of MAC protocols for wireless sensor networks and a number of representatives are presented in Section 3.2.

### 2.4.4 Networking Layer

In this section we review some requirements and features in topology control and routing. Both tasks are commonly performed at the networking layer (see Figure 2.2). Topology control is needed to manage connectivity in densely deployed sensor networks. Without topology control interferences and high redundancy in routing options might decrease performance. On the other hand, routing is required to forward data to a destination in a multi-hop network.

#### Topology Control

In order to provide redundancy and to support accurate sensing, sensor nodes are densely deployed in wireless sensor networks. Consequently, sensor nodes have many neighboring nodes. Besides the advantage of redundancy, this poses severe burdens on MAC and routing protocols. With a rise in network density, interferences and topology changes increase. Moreover, additional routing options are possible, which makes routing management more complex.

To overcome these problems, the application of topology control mechanisms has been proposed. The idea is to optimize the number of network nodes that are

required to guarantee connectivity. Basically, this can be achieved by two kinds of approaches [60]:

- **Identifying redundant nodes**: Redundant nodes are identified and temporary liberated from any communication need. These nodes can temporarily turn off their radios, which further conserves energy. Roles can periodically be changed. Thus, changing energy level distributions can be considered, which helps to further extend the network lifetime.

- **Transmission power control**: The number of neighboring nodes can be reduced by transmission power control. In addition to the prevention of interferences and congestion, energy can again be preserved. But, the average number of hops between source and destination nodes increases.

### Routing

In multi-hop networks routing data from a source to one or multiple destinations is needed. Intermediate nodes along a path have to decide to which neighbor they will forward a given data packet. Routing tables can be computed in advance or on-demand. Furthermore, they can contain information ranging from local neighborhood knowledge to global knowledge. If position information about the sensor nodes is available, this can be incorporated into routing too.

Two network properties are particularly relevant for routing in the context of this thesis: Network dynamics and the used communication pattern. The degree of network dynamics has a high impact on the design of topology control and routing mechanisms. If few topology changes are expected and communication is mainly from source to sink, a routing tree instantiated at the sink is adequate. On the other hand, if frequent topology changes occur or if multi-hop communication among network nodes is required, these approaches face difficulties due to their centralized nature. Accordingly, in these cases unrestricted fully distributed mechanisms are required. Distributed mechanisms provide the required means at the cost of additional communication and complexity. The state of the art in topology control and routing for wireless sensor networks is discussed in Section 3.3.

Another property of sensor networks is that the addressing scheme is typically changed from ID-based, e.g., used in MANETs, to data-centric. Instead of addressing individual nodes, the sensed data is of main interest. Content may be routed based on the kind of data, or data is stored distributed in the network according to the kind of data and the location where it has been sensed. In order to access this data, publish/subscribe methods and distributed hash table approaches have been proposed [37], [26], [121]. In order to organize tracking groups efficiently we have used an ID-based approach in our system design. However, global identifiers are not necessary, because individually addressing and connecting nodes separately from tracking group formation is not required.

### 2.4.5 Application Layer

In this section the top layer in the network stack is addressed (see Figure 2.2). As mentioned above sensor networks are used to monitor and optionally interact with the physical environment. Thereby two general classes of sensor systems are of interest: stream-oriented systems and event-oriented systems. Steam-oriented systems observe some physical phenomenon continuously. On the other hand, in event-oriented systems sensor nodes are programmed to become active upon presence of an event. Both approaches are introduced below.

#### Stream-Oriented Systems

Stream-oriented systems implement concepts of distributed data base systems. The focus is on long-term monitoring and data collection. Typical applications are environmental monitoring systems such as glacier and air pollution monitoring, or all kinds of agricultural and animal monitoring systems. Stream-oriented systems are well tailored to applications where a specific physical environment is continuously monitored. The networks are typically rather static and the implemented queries are periodically executed. The query syntax is similar to the syntax of SQL. A typical query might look as follows [93]:

**SQL-like Query:**

**SELECT AVG**(noise_level), room **FROM** sensors;
**WHERE** floor=5;
**GROUP BY** room;
**HAVING AVG**(noise_level) $>$ threshold;
**SAMPLE INTERVAL** 90s;

Stream-oriented systems support the generation and download of queries such as the one listed above onto the sensor nodes. These queries are then periodically executed on the sensor nodes that are matched by the query. In the example above reports containing the average noise level in the rooms located on the fifth floor of a building would be generated every 90 seconds.

Stream-oriented systems typically deploy routing trees in the sensor network. The trees are constructed according to the sensing task. New queries are downloaded onto the sensor nodes over these trees and the sensor readings (reports) are routed to the base station over these trees. In-network processing and aggregation are optionally applied at tree nodes to reduce communication load (aggregation is done in the example in Figure 2.1).

Stream-oriented systems have an intrinsic advantage in predicting system state and network load. Due to periodic sensing and the implemented routing tree, the network load is constant and can thus easily be monitored and controlled. Moreover, the query language provides aggregation primitives, which can be applied in-network at tree nodes without additional management costs. Collaboration and

arbitrary communication among the network nodes is barely required in stream-oriented applications due to the implemented routing tree. On the other hand, randomly occurring, unexpected events are difficult to be included into stream-oriented systems.

### Event-Oriented Systems

Unlike stream-oriented systems, event-oriented systems are useful if unexpected sporadic environmental events have to be detected and processed. Typical applications are surveillance (e.g., building, disaster), security, and military. In such applications periodic sensing is an overhead. Moreover, stream-oriented applications might not be able to deal with abruptly evolving events in a timely manner. In contrast, event-oriented systems are designed to meet the specific event detection requirements. In event-oriented applications events are generally detected and monitored on-demand and fully distributed. Because of the lack of a fixed infrastructure, energy can be saved as long as no critical phenomenon is detected. On the other hand, the lack of an infrastructure requires collaboration and negotiation, which provides flexibility but also imposes complexity. All system functionality is implemented according to the specific application requirements.

```
Event-oriented task:

if event detected
   Negotiate state with neighbors;
   if state == LEADER
      Request event information from neighbors;
      Generate event report based on collected data;
   else if state == MEMBER
      Report requested information to leader;
   end
end
```

A task executed by sensor nodes of an event-oriented system is described in pseudo code above. Upon detection of an event, sensor nodes are instructed to organize themselves. Therefore, negotiation among the sensor nodes is required. This is not required in streaming-oriented systems. As soon as the roles are assigned to the sensor nodes, the nodes perform their respective task.

Because events occur infrequently in event-oriented applications, communication occurs burst-like and is more difficult to predict or control. Any kind of aggregation is application-specific and needs to be implemented by the system designer. Because events and their handling are modeled from scratch, no common aggregation functionality is provided.

## 2.5 Conclusions

In this chapter a general overview of wireless sensor networks, their benefits, their challenges, and their applications has been given. The main focus was on topics which are also relevant for the work done in this thesis. It has been shown that wireless sensor networks extend conventional sensor systems by providing the means to monitor a phenomenon of interest directly, close to the location of incident. Moreover, no cables are needed, which lowers deployment costs considerably. These properties open a new range of applications such as glacier monitoring that would be difficult to address with conventional systems. On the other hand, wireless sensor networks impose a number of challenges such as error-proneness, accuracy, self-organization and energy consumption. The need to preserve energy is probably the most limiting factor in wireless sensor networks design. In particular if long-term operation is required, energy-efficient algorithms are mandatory.

In our own work we focus on the detection and processing of unexpected events. This context poses severe constraints on delays and requires real-time processing and reporting. Accordingly, distributed storage of sensed data and stream-oriented mechanisms are not best suited. In contrast, event-oriented distributed tasking such as monitoring group formation, on-demand data aggregation and reporting are required. Our event detection system has been designed in order to meet the requirements discussed in this chapter. Particular focus has been on the trade-off optimization between needed accuracy and energy savings.

The next chapter discussed related work relevant for the work performed in this thesis.

Chapter 3

# Related Work

In this chapter we give an overview of related work that is important for the investigations carried out in this thesis. The sections are organized according to the chronology of the subsequent chapters. An event monitoring framework requires functionality on different network layers. Accordingly, both related work and chapters are organized in a bottom-up manner starting at the MAC layer. The protocol stack is traversed up to the application layer.

## 3.1 Introduction

The related work discussed in this chapter is shortly summarized in the following. Current state-of-the art in medium access control is presented in Section 3.2. Focus is on energy-efficient contention-based MAC protocols. Sensor node synchronization and coordinated sleeping on the MAC layer are furthermore addressed.

On the network layer topology control and routing are addressed. Relevant related work is presented in Section 3.3. The focus is on connected dominating sets used to establish routing backbones. State-of-the art algorithms of connected dominating sets mainly try to minimize the number of backbone nodes. However, because the backbone guarantees routing, nodes that are not elected into the backbone can shut-down their radios as long as no events are present. Thus, considerable energy can be saved in wireless sensor networks.

The state-of-the art for the application layer is discussed with respect to the application-specific features of our system (see Figure 1.1). Related work in event detection and tracking is presented in Section 3.4. Centralized and more distributed algorithms are discussed. The state-of-the art for event localization and signal strength estimation is presented in Section 3.5. Different event classification and anomaly detection mechanisms are presented in Section 3.6. The classifiers and anomaly detectors include simple threshold-based approaches and more complex algorithms based on statistics and/or reasoning. The classification of both, discrete and continuously present events is addressed. A number of typical event monitoring applications are presented in Section 3.7. Finally, Section 3.8 presents the two sensor node platforms that have been used during the development of this thesis.

## 3.2 Medium Access Control

In this section different relevant medium access control (MAC) approaches are presented. MAC protocols can mainly be divided into contention-based protocols and protocols which divide the medium access into contention-free time slots that are assigned to the network nodes [142]. These time-division-based MAC protocols impose complex slot handling and scale worse than contention-based protocols. Moreover, they are more dependent on very precise synchronization. For these reasons we focus on contention-based protocols in our work. Contention-based MAC protocols can be further divided into synchronous and asynchronous MAC protocols [140]. Synchronized contention-based MAC protocols implement low duty cycles that need to be synchronized. On the other hand, asynchronous contention-based MAC protocols implement a type of preamble sampling.

### 3.2.1 Synchronized Contention-Based Medium Access

Different synchronized contention-based MAC protocols have been proposed for usage in wireless sensor networks. In order to preserve energy, low duty cycles are implemented. This means network nodes follow periodic listen/sleep cycles. In the listen periods they wake up, synchronize with neighbor nodes and take part in communication in case some data is pending for transmission. If no data traffic is indicated, the nodes fall back to sleep after the listen period.

#### Medium Access Control with Coordinated Adaptive Sleeping (S-MAC)

S-MAC [173], [172] is an energy-efficient contention-based MAC protocol for wireless sensor networks. It is based on low duty-cycles and requires the exchange of synchronization (SYNC) messages to synchronize the listen/sleep schedules of the nodes. Every node maintains its own listen/sleep schedule. These schedules are synchronized whenever possible in order to reduce control traffic overhead. Nodes maintaining the same listen/sleep schedule build virtual clusters. New sensor nodes listen to the wireless medium for a certain amount of time to overhear and adapt existing schedules. If no SYNC message has been received during this period, a node chooses its own schedule. Any subsequently overheard different schedule is adapted too. Thus, virtual clusters are interconnected. All interconnecting nodes follow multiple schedules, i.e., the schedule of each cluster they are a member of, and accordingly consume much more energy than normal cluster nodes. Apart from virtual clustering, the SYNC messages are also used to adjust clock drifts between network nodes.

In addition to virtual clustering and normal RTS/CTS procedures, S-MAC implements adaptive listening and message passing. Every transmitted RTS/CTS message is extended with a network allocation vector (NAV) value that indicates the duration of the following data transmission. Thus, all nodes which do not take part in the current communication are enabled to turn off their radio until the ongo-

ing communication is finished. The transmission of long data messages imposes increased error rates. Therefore, such messages are fragmented. In message passing all resulting fragments are transmitted in a burst, with only one exchange of RTS/CTS in the beginning. Thus, the transmission of RTS/CTS packets for every single fragment can be avoided.

### An Adaptive Energy-Efficient MAC Protocol (T-MAC)

T-MAC [142] is a traffic-adaptive energy-efficient MAC protocol for wireless sensor networks. Like S-MAC, it implements virtual clustering and an RTS/CTS procedure. Unlike S-MAC, T-MAC supports the adjustment of listen periods in dependence of the pending data traffic. This adaptive listening is depicted in Figure 3.1. The gray bars indicate active radios.

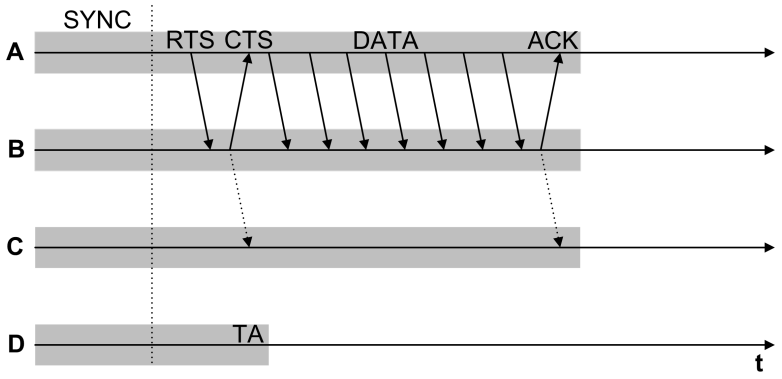



**Figure 3.1:** Adaptive listening in T-MAC.

T-MAC introduces an activation time TA that covers the synchronization period and the transmission of one RTS/CTS exchange. TA determines the minimum amount of idle listening. If for the duration of the TA no RTS or CTS has been overheard by a node, e.g., node D in Figure 3.1, the node (D) immediately goes to sleep. Nodes A and B stay awake because of the data transmission. Unlike S-MAC, node C performs no adaptive listening in T-MAC. Thus, throughput is improved. In any case, due to its traffic-aware operation, T-MAC would gain less from adaptive listening than S-MAC. The duration of the TA has been designed to span over some short contention period, the transmission of one RTS/CTS exchange period and a random backoff. The TA of T-MAC is considerably shorter than the static listen period implemented in S-MAC. Accordingly, T-MAC performs particularly well in scenarios with little and irregular traffic.

To increase throughput, T-MAC provides a future request to send (FRTS) mechanism. With FRTS, nodes that lose contention can inform their neighbors about pending transmissions. If node C in Figure 3.1 had a message pending for node D, it would inform node D about its pending data transmission (with a FRTS), before waiting the data exchange between nodes A and B. Of course, nodes A and B wait

for a possible FRTS from node D before starting data exchange. Having overheard the FRTS, node D would abstain from going to sleep.

### Traffic Aware Energy Efficient MAC Protocol (TEEM)

The Traffic Aware Energy Efficient MAC (TEEM) protocol [138] is another optimization of S-MAC. TEEM again reduces the overhead introduced by the fixed length listen schedules in S-MAC. In TEEM, both SYNC and RTS messages are combined into a single message, which is sent prioritized if data is pending for transmission. Therefore, the listen period is divided into a SYNC_DATA and a SYNC_NODATA part. If some data is pending for transmission, the respective nodes inform their neighbors about those transmissions in the SYNC_DATA period. SYNC packets are used in the SYNC_DATA period to inform neighbors about an upcoming data transmission in a RTS-like way as well as to synchronize network nodes. The receiver of the control message immediately responds with a CTS message and the data transmission can be started. On the other hand, if no data is pending for transmission, the SYNC_DATA period expires without any transmission of a control message and all network nodes contend to transmit their synchronization message, which has no RTS functionality this time, in the SYNC_NODATA period.

With the proposed mechanism the listen period of S-MAC can be reduced and energy can be saved. Unlike T-MAC, TEEM does not provide any future request to send mechanism. TEEM introduces higher delays and smaller throughput compared to S-MAC and T-MAC.

### Demand-Wakeup MAC

Demand-Wakeup MAC (DW-MAC) [139] is a recent synchronized contention-based MAC protocol that is based on low duty-cycles and traffic-adaptive wake-up periods. Like the other approaches DW-MAC organizes neighboring nodes into virtual clusters. Accordingly, the synchronization mechanism has not been altered compared to S-MAC, T-MAC, and so on. However, unlike other approaches DW-MAC does not use RTS/CTS to allocate the channel, but implements an on-demand scheduling mechanism. Nodes negotiate subsequent data transmissions in the listen period. Fix data transmission slots are assigned to receiver nodes. Thus, overhearing and collisions can be avoided while minimizing the duty cycle. With DW-MAC, channel allocation and channel usage have been shown to be more efficient compared to previous protocols.

### Pattern MAC

In Pattern MAC (P-MAC) [176] listen/sleep schedules are determined adaptively. Thus, drawbacks of fixed cycles, of algorithmically limited throughput and of increased energy consumption under light network load are diminished. The schedules are determined based on the own traffic of a node as well as of the traffic of its

neighbors. Local traffic patterns are learned from communications observed by a node. These traffic patterns are used to determine the duration of the sleep period of a node. The pattern is represented as a string of bits which indicates the listen/sleep plan for several cycles. The sleep time is exponentially increased in presence of low traffic. The mechanism is somewhat similar to the slow start mechanism of TCP. Accordingly, if no traffic is indicated in the neighborhood of a node, the node can go to sleep for a long time.

P-MAC has been tested with constant bit rate communication along a single path. This scenario is tailored to P-MAC. The performance and adaptivity under variable traffic patterns remains to be shown. A key problem of PMAC is that all nodes that are not located along a path are in a long-sleep state, which introduces high delays. The design of PMAC is tailored to unicast traffic.

### Synchronization of Listen/Sleep Schedules

Nodes that run a synchronized contention-based MAC protocol follow periodic listen/sleep schedules. Nodes sharing the same schedule are virtually organized into clusters. In a sensor network multiple clusters might evolve. To support communication between these clusters, border nodes that interconnect them are required.

The problem of virtual clustering, i.e., of coexisting schedules, has been addressed in [85]. Experiments have shown that already in a multi-hop network consisting of 50 nodes, which run S-MAC, up to four different virtual clusters have evolved. Moreover, it has been shown that border nodes had to listen to up to three different schedules. Thus, border nodes have higher average energy consumption than normal cluster nodes. Because the synchronization procedure is similar for all synchronized contention-based MAC protocols, all these protocols would share the behavior of S-MAC. Virtual clustering is illustrated in Figure 3.2.

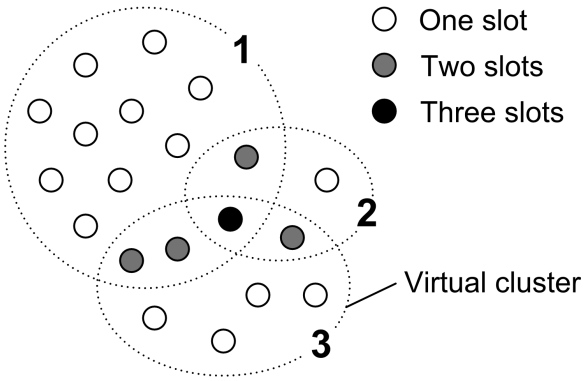



**Figure 3.2:** Drawback of virtual clustering.

The gray and black nodes in Figure 3.2 operate as border nodes (gateways) between the clusters and have to listen to multiple schedules. Accordingly, these nodes sleep less and their batteries deplete sooner. If this happens, network con-

nectivity might be broken and the network might be out of order, even though sufficiently many working network nodes might still exist. Therefore, it is desirable to avoid virtual clustering. In [85] it has been shown that surprisingly many nodes follow multiple schedules. Mainly the temporary unavailability of communication links and the presence of gray regions [175] in radio communications, i.e., of effects due to strongly varying radio ranges, have been identified as reasons.

In [85] an additional schedule age has been introduced to solve the problem. The authors motivate that different schedules must have entered the network at different time points and thus have different ages. The schedule age is announced in the SYNC message. Over time all nodes converge towards the oldest schedule in the network. To prevent network partitions all other schedules need to be temporary stored too. The maintenance and distribution of the schedule age requires additional information. In our own work we will show that no schedule age is needed because local schedule consistency is sufficient.

### 3.2.2 Asynchronous Contention-Based Medium Access

A second group of protocols provides asynchronous contention-based medium access. Most of these protocols implement preamble sampling. Unlike the protocols proposed so far, they do not require any synchronization, but send long preambles in order to reach neighboring nodes, which might be asleep. The minimum size of the preamble is determined by the maximum length of the sleep cycle. An example of preamble sampling is shown in Figure 3.3.

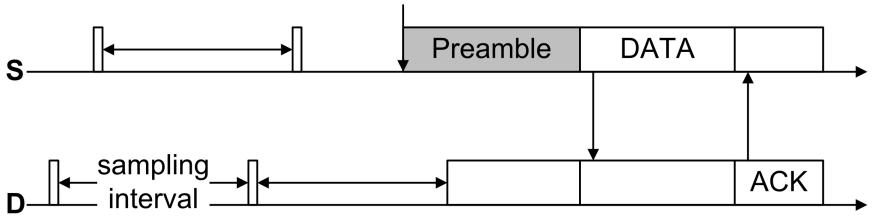



**Figure 3.3:** Preamble sampling and data transmission.

Both, sender S and destination D periodically wake up to sense the carrier. Upon arrival of a message, S starts to send a preamble with the message attached to it. D overhears this preamble, stays awake and receives the message. The transmission is confirmed with an acknowledgment message by D.

WiseMAC [33] is a contention-based MAC scheme based on preamble sampling. Each node periodically wakes up for a very short period in order to listen if some preamble is being sent. To determine whether a preamble is being sent or not, the received signal strength indicator (RSSI) is measured. The sampling is periodic with a fixed length sampling interval. If the medium is occupied, any observing node remains awake until the data packet is transmitted. Otherwise, every node goes immediately back to sleep after the sampling period. If a node is not addressed in the data packet, it switches to sleep state too. The preamble length can

be minimized by learning the sampling schedules of the neighbors. This information is included into acknowledgment messages. With this knowledge a sender is able to predict the wake up time of a neighbor node and therefore it knows when to start sending the preamble. In Figure 3.3 node S would accordingly start to send its preamble close to the next sampling period of node D. The minimum length of the preamble is $Tp$, where $Tp = min(4 \cdot \Theta \cdot L, Tw)$. $Tw$ is the sampling period, $\Theta$ is the tolerated frequency and L is the predicted time point of transmission.

B-MAC [111] is another contention-based MAC protocol that uses adaptive preamble sampling to reduce duty cycles and to minimize idle listening. B-MAC is a lightweight asynchronous MAC protocol. It uses a Clear Channel Assessment (CCA) to determine whether the channel is clear or not. The CCA is similar to the RSSI measurements done by WiseMAC. B-MAC achieves low duty cycles by periodic channel sampling, which is called Low Power Listening (LPL) in B-MAC. The preamble length is chosen to be equal to the sampling interval. The duration of the preamble length has not been minimized in B-MAC. B-MAC provides minimal services and is therefore highly configurable. RTS/CTS procedures can for example be implemented as a system service. Thus, solutions can be implemented which are tailored to specific applications. The protocol overhead is small.

X-MAC [13] is another protocol based on preamble sampling that optimizes the preamble length. X-MAC introduces a preamble exchange scheme which transmits a series of short preambles. Each of these preambles contains information about a receiver. The proposed mechanism minimizes overhearing costs at non-target receivers.

In RI-MAC [140] receiver nodes announce their availability by beacon messages. Based on the reception of such a beacon, a sender node transmits its pending data to the receiver. Accordingly, RI-MAC substitutes the sender-initiated preamble transmission with receiver-initiated beacon messages. RI-MAC has shown to outperform preamble-based MAC protocols in a wide range of traffic loads.

All asynchronous MAC protocols achieve low duty cycles. However, they require the exchange of preambles or beacons and support broadcast operations poorly. In addition to broadcast support, periodic synchronization messages can be further used to learn neighborhood information without additional control traffic.

### 3.2.3 Time Division Multiple Access

Time Division Multiple Access (TDMA) based MAC protocols also require the exchange of periodic SYNC messages in order to operate. However, unlike contention-based protocols, the operation of TDMA-based protocols is based on the concept of clusters. In general they require a cluster leader which allocates slots to its cluster members. Thus, contention-free medium access can be guaranteed in every slot. On the other hand, TDMA-based MAC protocols require complex management, very precise time synchronization and do not scale well. Therefore, contention-based protocols have been considered for our work.

### Lightweight MAC Protocol (LMAC)

LMAC [143] is a TDMA-based protocol, where each node possesses exactly one slot in a frame. A frame is divided into 32 equal slots, which can be spatially reused. Within a slot the owner node can communicate collision-free. All sensor nodes shortly awake in each slot to overhear the synchronization message of the slot owner. This message furthermore advertises the destination address of a data packet. A node that is not listed as a receiver of a data message, can immediately go to sleep for the rest of its current slot. In LMAC a node can occupy exactly one slot. The spatial reusage of time slots is organized such that collisions are prevented, i.e., such that the slot-occupying nodes cannot interfere. If more nodes have to be supported, more slots are needed. Thus, the frame length grows, which leads to longer delays, because nodes can only communicate in their slots. Based on these requirements, the local two-hop neighborhood of LMAC is restricted to 32 nodes.

### A MAC Protocol for Long-Term Applications

A-MAC [88] is a recent hybrid MAC protocol that is based on TDMA. It uses an advertisement mechanism to avoid collisions and to minimize overhearing and idle listening. Unlike LMAC, A-MAC implements the notification of future data transmissions. Thus, throughput and delay can be optimized. Apart from this notification of future data transmissions, A-MAC operates similar to LMAC. In addition, the protocol provides two modes that consume different energy to better support application-specific requirements.

### Traffic Adaptive Medium Access Protocol (TRAMA)

TRAMA [119] is a collision-free MAC protocol that uses a distributed election scheme to determine time slots. TRAMA distinguishes between contention-based random access slots which are used for signaling, and scheduled access periods that are used for collision-free data exchange. TRAMA consists of three components: The Neighbor Protocol (NP), the Schedule Exchange Protocol (SEP), and the Adaptive Election Algorithm (AEA). NP propagates one-hop neighborhood information to the signaling slots. Thus, the local two-hop neighborhood can be learned. With the SEP protocol traffic-based schedule information is maintained among neighbors. SEP packets are exchanged during the scheduled access periods. AEA selects transmitters and receivers according to the information obtained from NP and SEP. TRAMA supports multicast communication by using a bit-mask vector which contains the one-hop neighborhood information. Whenever nodes are not scheduled by the AEA protocol, they switch to sleep state. TRAMA introduces high delays. On the other hand, good throughputs can be achieved.

## 3.3 Routing and Topology Control

In this section we give an overview of the state of the art in routing and topology control. Topology control mechanisms have been introduced to provide network connectivity, while trying to minimize the number of active network nodes. Thus, energy can be saved. Moreover, congestion and collisions can be prevented. The focus is this section is on connected dominating sets (CDS), because they can be implemented fully distributed, impose manageable complexity and can provide both routing and topology control.

### 3.3.1 Routing Protocols for Wireless Sensor Networks

In this section we give a brief overview of the state of the art of routing in ad-hoc and wireless sensor networks. Numerous routing protocols have been proposed for mobile ad-hoc networks (MANET). Some of these protocols have been adapted to wireless sensor networks too. The protocols can mainly be divided into proactive protocols, where routing paths are computed in advance such as OLSR [25] and DSDV [110], and in reactive protocols such as AODV [108] and DSR [59], where routing paths are computed on demand only if some data is pending for transmission. Finally, there are location-based protocols, e.g., GPSR [61], BLR [49], which presume knowledge of position information in their routing decisions. In particular location-based routing protocols have also gained a lot of attention in wireless sensor networks. This is not surprising, because knowledge of location information eases routing considerably and is often per se required in the context of wireless sensor networks.

With the development of wireless sensor networks a new property evolved which has a high impact on the design of routing protocols, namely the paradigm of data-centric operation. In MANET network nodes are typically addressed by address identifiers such as an IP. On the other hand, IDs of sensor nodes are in general less important than the physical context of the nodes, which depends on their geographic location. Accordingly, the addressing scheme in sensor networks has changed from ID-based to data-centric. This means that packets are no longer routed according to a destination ID, but according to the context of the searched data. The search direction is either determined by queries or by negotiation. Typical query-based approaches are Directed Diffusion [56], [57], Rumor Routing [10] and GHT [121]. A typical negotiation-based routing protocol is SPIN [48]. The discussed routing protocols support node-to-node communication, whereas our application only needs source-to-sink communication.

### 3.3.2 Topology Control in Wireless Sensor Networks

Connected dominating sets, which will be discussed in the next section, establish connected network backbones that can be used for routing and/or topology control. In this section we give a short overview of alternative state of the art in topology

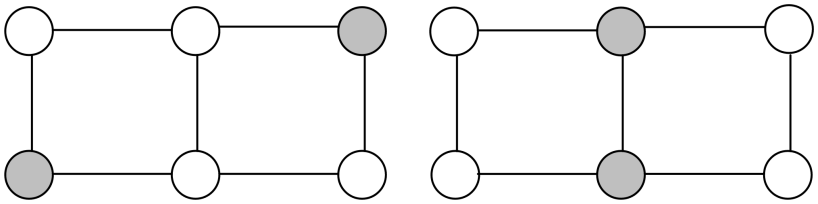
control. As mentioned before, topology control has been introduced to save energy in densely deployed sensor networks, either by temporarily disconnecting nodes that are not required to guarantee routing, or by transmission power control. In general it is desirable to change the roles of the nodes from time to time to achieve a uniform energy load distribution in a network over time.

The local identification of redundant nodes, i.e., of nodes that are temporarily not required to guarantee connectivity, in a network graph $G$ has been addressed by computing the Gabriel Graph (GG) [40] of $G$. A similar approach computes the Relative Neighborhood Graph (RNG) [141] of $G$. A third approach to minimize the number of nodes in a graph is by using Delaunay Triangulation [29]. More approaches that generate subgraphs have been proposed in [83] and [82]. In [80], [84] and [62] the computation of distributed Minimum Spanning Trees (MST) has been proposed in order to identify and temporary turn off redundant nodes. In the cone-based topology control [136] nodes determine a subset of neighbor nodes as communication partners by beacon transmissions with iteratively increasing transmission power. Nodes that are not elected by the mechanism can go to sleep. In COMPOW [102] topology control has been addressed by using different routing protocols operating on different power levels.

### 3.3.3 Connected Dominating Sets

Connected dominating sets can be established in a distributed manner by exploiting neighborhood information. As presented in the introduction we exploit the synchronization messages exchanged by synchronized contention-based MAC protocols to learn neighborhood information. Accordingly, a routing backbone based on connected dominating sets can be implemented directly on the MAC layer. Thus, no additional control traffic is required. Apart from routing support, additional energy can be saved by temporarily turning of the non-backbone nodes. This is possible because routing and thus network connectivity is guaranteed by the backbone. Relevant related work in connected dominating set research is discussed in the following. First some preliminaries are presented. A connected dominating sets (CDS) is characterized as follows: A dominating set (DS) of a graph $G = (V, E)$ is a subset $V' \subset V$, where each node in $V - V'$ is adjacent to some node in $V'$. A CDS is a dominating set which builds a connected subgraph of $G$. Two simple examples of a DS and a CDS, respectively, are depicted in Figure 3.4. The gray nodes are members of the respective dominating set.

To minimize the number of backbone nodes it is desirable to find a minimum connected dominating set (MCDS) of $G$. Finding an MCDS is however NP-complete [24]. Consequently, heuristics are applied. In our work we propose two fully distributed approaches. In Figure 3.4 the CDS on the right side also builds an MCDS. If any of the remaining white nodes were colored gray, the CDS would still be a CDS, but it would no longer be an MCDS. As mentioned above, the computation of the MCDS is in general very cost intensive and requires global knowledge and small network sizes.



**(a)** Dominating Set (DS).

**(b)** Connected Dominating Set (CDS).

**Figure 3.4:** Examples of a DS and a CDS.

The main goal of our approach is to extend network lifetime rather than to minimize the CDS. Therefore, the energy level of a node is used in the election procedure too. However, first some related work that mainly focuses on the minimization of the CDS is presented. Most work is from research in mobile ad-hoc networks (MANET), where energy is a less important performance criterion than in wireless sensor networks.

### CDS Based on Pruning Rules

The algorithm proposed in [167] first determines a CDS consisting of all nodes that have at least two non-adjacent neighbors. This initial CDS is reduced by applying two pruning rules. The resulting connected dominating sets are depicted in Figure 3.5. The initial CDS that contains all nodes with unconnected neighbors is shown on the left (black nodes). On the right the CDS after having applied both pruning rules is shown (the remaining two black nodes).

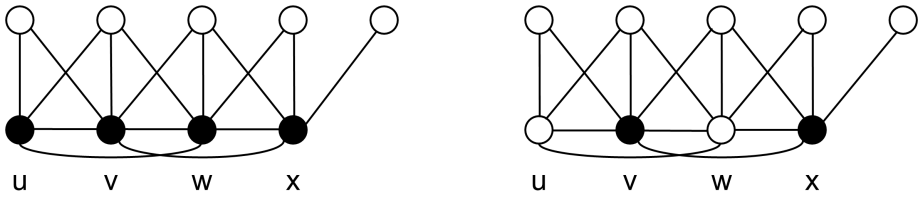



**Figure 3.5:** CDS with pruning rules.

The first rule removes nodes from the set which are completely covered by other nodes in the CDS. Node u is removed by this rule. The second rule removes nodes that are fully covered by the neighbor sets of two neighbor nodes. Node w is removed from the CDS according to the second rule. The algorithm needs two-hop neighborhood information and can perform poorly in specific networks [159].

In [166] the pruning rules have been adapted to consider the energy levels of the nodes in the setup phase. Instead of taking the link degree into consideration in

the pruning process, the authors propose to use the energy levels of the nodes. The algorithm is thus able to consider energy distributions in the network.

### Connecting a Maximum Independent Set

With the algorithm proposed in [159] as a first step a maximum independent set (MIS) is constructed. This is a dominating set which contains only nodes that are two-hops away from each other. In a second step the MIS nodes are connected by electing a set of appropriate intermediate nodes. The functionality is depicted in Figure 3.6.

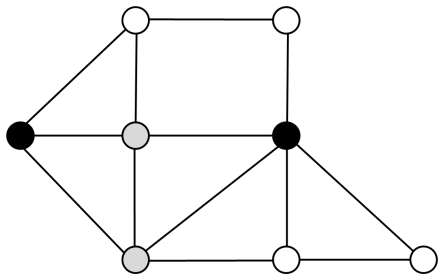


**Figure 3.6:** Connecting a maximum independent set.

The two black nodes in Figure 3.6 build a maximum independent set. To achieve connectivity this MIS needs to be connected. In [159] a heuristic is proposed which chooses the connecting nodes effectively, i.e., by minimizing the number of resulting nodes in the backbone. In Figure 3.6 one of the two gray nodes would be chosen into the final CDS. The algorithm is rather complex, time-consuming and static.

### A Simple Timer-based CDS Construction Mechanism

A simple greedy procedure to establish a CDS has been proposed in [177]. Again, each node knows its local neighborhood by the exchange of beacons. The decision whether to join the CDS or not is done by a simple greedy procedure that evaluates the number of remaining uncovered nodes when a timer expires. The algorithm is initialized by a dedicated node. Each node entering the CDS broadcasts a message containing its list of neighbors. The beacon messages of the IEEE802.11 MAC protocol are used. Each receiver marks the common neighbors as dominated and sets a timer according to the number of remaining unmarked neighbors:

$$\Delta T = T_{max} \cdot \frac{1}{(\textit{number of uncovered neighbors})^{\alpha}}$$

The parameter $\alpha$ weights the impact of the number of uncovered neighbors. Large values of $\alpha$ cause very short time outs for nodes that have many uncovered neighbors. If the timer expires, the node enters the CDS. As soon as a node has no

unmarked neighbors left, it terminates the timer and enters a non-backbone state. The timer is reset whenever a message is received. The algorithm is simple and fast, but approximates the MCDS rather poorly. Moreover, energy levels have not yet been considered in the decision process.

### 3.3.4 Multipoint Relaying Protocol

In our work we propose mechanisms to construct connected dominating sets. Two of them are related to the Multipoint Relaying protocol (MPR) [116], which is used for efficient broadcast in the Optimized Link State Routing (OLSR) [25] protocol. MPR requires knowledge of two-hop neighborhood information. Based on this information, subsets of one-hop neighbors are forced to rebroadcast given data packets. These forwarding nodes are called Multipoint Relays.

Since a Multipoint Relay knows its local two-hop neighborhood, it can choose its most efficient one-hop neighbors as subsequent Multipoint Relays. The set of all Multipoint Relays establishes CDS. The Multipoint Relay set of a given Multipoint Relay $x$ is calculated according to the following algorithm:

**Selecting the Set of Multipoint Relays**

1. For each neighbor $y$ of $x$ calculate the number $D(y)$ of two-hop neighbors $z$ that are connected to $x$ over $y$.
2. Add to the Multipoint Relay set those $y$ that provide exactly one link to a two-hop neighbor $z$. Remove all $z$ that are now covered over $y$ from the two-hop neighbor list.
3. While two-hop neighbors exist that are not yet covered by a node in the Multipoint Relay set repeat:
   - 3.1. Compute the coverage of each $y$, i.e., compute the number of remaining two-hop neighbors $z$ connected over $y$ that are not yet covered by a node in the Multipoint Relay set.
   - 3.2. Select the $y$ as Multipoint Relay that provides the largest coverage. If multiple $y$ show the same coverage, select the node with highest $D(y)$. Remove all $z$ that are now covered from the two-hop neighborhood list.

#### An energy-efficient coordination algorithm for topology control (Span)

Span [20] is a distributed algorithm where nodes make local decisions whether they join a forwarding backbone as a coordinator or if they are not required to support routing and accordingly can go to sleep. The resulting backbone is a connected dominating set. The decision process requires knowledge of the local three-hop neighborhood information, which is collected by exchanging the one- and part of the two-hop neighborhood information of each node in its vicinity. Considering the two-hop information, only the exchange of the according coordinator information is required.

The algorithm supports efficient routing and elects redundant nodes effectively. On the other hand, the requirements in terms of storage and control message communication are high. This makes the algorithm less applicable to sensor networks.

### Geography-informed Energy Conservation for Ad-hoc Routing

In GAF [169] nodes form virtual clusters, where redundant nodes, i.e., nodes not required for routing, are temporarily disconnected from the network. The virtual clusters are determined based on geographical information. The network is divided into a grid of cells. GAF ensures that within each cell at least one active sensor node is always present. This active node is needed to guarantee and provide routing. Accordingly, GAF also establishes a connected dominating set. GAF utilizes geographic location information, and divides the network into fixed square cells. Each node in GAF has to provide a location service (e.g., GPS). The algorithm is efficient and simple. However, location information is presumed, which can be a limiting factor. GPS for example is not available indoors and other solutions might lack accuracy.

### Low Power Media Access (ASCENT)

ASCENT [18] determines redundant nodes by local negotiation procedures. Each node makes its decision whether to stay active or not only based on local connectivity information and measured packet loss. In ASCENT active nodes are adaptively selected among all nodes in the network. Active nodes are awake all the time and guarantee multi-hop packet routing. All other nodes are passive and turn off their radio. They only wake up periodically to check whether they should become active or not.

ASCENT requires high node density. Initially only some nodes are active. These nodes route all data traffic. Upon detection of poor communication links, i.e., high packet loss is indicated, the according active node(s) start to send help messages. Passive nodes overhearing these help messages become active and help in forwarding the data traffic. This procedure continues until the number of active nodes is stable, i.e., no poor communication links are indicated anymore. ASCENT provides no mechanism to release active nodes from their role. Thus, the number of active nodes grows over time. Moreover, because active nodes cannot be replaced, the active nodes will run out of energy much faster than passive nodes. Accordingly, no energy load distribution mechanism is provided. The active nodes in ASCENT build a connected dominating set.

## 3.4 Event Detection and Tracking

In this section the state of the art in efficient source detection and tracking is discussed. The contributions are mainly from the networking and distributed communications research field. Methods for tracking group formation and maintenance

are proposed. The focus is on efficiency, network lifetime and communication load minimization. In most cases, tracking groups are dynamically established. Such local group organization keeps the communication costs low. The downside is that accurate event localization and classification, which are typically based on collaborative signal processing (CSP) and require detailed information to formulate and solve the according problems, are given less insight.

### 3.4.1 Target Localization in Distributed Sensor Networks

In [179] a sensor node deployment and target localization framework has been proposed. After deployment, all network nodes are organized into local clusters, which remain static. Upon detection of an event, a cluster head elects a subset of cluster members to provide it with detailed information about the event.

The target tracking application consists of a two-step communication protocol. Upon detection of a target (an event), the respective detecting sensor node notifies its cluster head about the target by a very short control message. In this message the presence of an event is indicated by only one bit. The cluster head thereafter queries a subset of its cluster members to obtain more detailed information about the target. The subset of nodes that have to be queried is determined based on a score-based ranking algorithm. The score-based ranking algorithm itself is based on a detection probability table, which is maintained by the cluster head and contains a detection probability for each cluster member. The goal of this message exchange procedure is to minimize communication costs.

The static cluster formation is inflexible. The probability of target detection by multiple clusters is increased close to cluster boundaries. Accordingly, either the number of reporting clusters could become high or additional negotiation among the cluster heads is required. Both effects increase network load. Localization and classification algorithms have not been addressed. However, they could be supported depending on the kind of information requested by the cluster heads.

### 3.4.2 SensIt: Region-Based Detection and Tracking of Targets

In [78], [120] the authors propose region-based CSP for target detection, tracking and classification. The regions are dynamically created and deleted by a location-centric application programming interface (API). The API is motivated by the message-passing interface standard [101]. The sensor network is divided into dynamically established cells. The diameter of the cells depends on the target velocity. Each cell contains numerous nodes, whereof at least one node is leader. The leader node collects and processes the data received from the other nodes in the cell. Subsequent cells are initialized if the moving target leaves the current cell. The monitoring and tracking of a single target is illustrated in Figure 3.7.

The target enters initial cell A. The leader node(s) in cell A predict the tracking path of the moving target based on statistical methods. If the target moves towards the boundary of the active cell, adjacent cells (regions) are activated according to

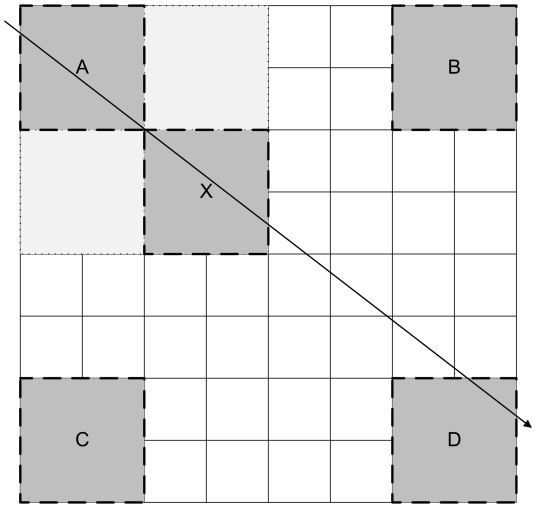



**Figure 3.7:** Grid creation and handover in SensIt.

the estimated moving direction. As soon as the target enters activated cell X, X becomes active. Active regions become inactive when no event has been present for a given amount of time. The message passing scheme involves rather high communication costs. Moreover, the handover procedure that activates all cells adjacent to an active cell is expensive too.

Similar approaches have been proposed in [12], [11]. Here, the authors have proposed a distributed cooperation framework instead of the location-centric API for the target tracking. The work is also part of the SensIt project. The observation area is again divided into grids. Publish/subscribe mechanisms are used for communication within the grids. One member of the grid is chosen as leader and aggregates the observations from the members and reports the resulting information. Cell handover is performed similar to [78].

### 3.4.3 Event Detection and Tracking with Consensus

In [71] the self-organization of event observing groups is identified as key problem of distributed event detection and tracking. The main goal is to avoid the establishment and maintenance of multiple tracking groups for the handling of a single spatially-restricted event. Single groups are established based on consensus. Consensus itself is based on a quorum mechanism. Thus, the coexistence of event tracking groups can be prevented. Hence, only one tracking group reports to the base station. Coexisting events are supported as long as they are sufficiently disjointed in space.

To achieve consensus, the protocol requires multi-step negotiation among the event observing nodes. As soon as an event has been observed, each concerned sensor node sends a consensus request in a PROPOSE message. Neighbor nodes answer this message with an ACK message that includes their sensor readings.

Based on all collected sensor readings, a simple majority decision is performed. Thus, the most relevant sensor node is determined in a distributed way among all event observing nodes. The node with highest ranking makes the majority decision. The winning node informs all other nodes about its state with a DECIDE message. The communication costs of the protocol are rather high. Moreover, the protocol requires the sensing range to be at most half the communication range. Otherwise, the negotiation costs increase considerably, because two- and more hop communication would be required.

### 3.4.4 Event Detection Using Data Service Middleware

Another approach that supports distributed event detection in wireless sensor networks is DSWare [81]. DSWare supports data-centric storage by applying hashing functions to map data to physical nodes. To provide robustness, the data is replicated in multiple physical nodes which are mapped to a single logical node. Moreover, queried data is often cached on multiple sensor nodes along the routing path. Thus, communication costs can be saved. Apart from streaming-based data collection services, DSWare also supports event-based monitoring.

With DSWare an application specifies a compound event to collect relevant information from a certain geographical area. The event specification consists of a maximum detection range, a time interval, and a confidence function. This confidence function describes how the different measurements, collected from the sensor nodes in the monitoring area, are weighted and how the composition of the data is calculated. If the computed value exceeds a certain threshold, the event detection is significant and a report is sent to the base station. The context of events is modeled according to Finite State Machines (FSM), which requires system experts to design particular event-dependent FSMs. An event is modeled by the associated FSM, which is ultimately downloaded onto the sensor nodes of interest. Similar approaches have been proposed in [92] and [124]. GADT [38] models events that are observed with streaming-based systems according to Gaussian Abstract Data Types (GADT) in order to compensate for inaccuracies in the sensed data.

### 3.4.5 EnviroTrack: An Environmental Computing Paradigm

EnviroTrack [1], [90] is an object tacking middleware. Tracking objects are dynamically created and logically attached to selected external entities. Localization and classification cannot be supported without adaptations.

As soon as a moving object is detected by some node, tracking groups are dynamically established. Group leaders are elected based on contention and organize their tracking group by periodically broadcasting heartbeat messages. Upon sensing of an event, i.e., when the configurable sensing function of EnviroTrack fires, each concerned sensor node sets a random timer (contention). When this timer expires, the respective node appoints itself as leader and immediately starts to broadcast heartbeat messages. Each node that overhears a heartbeat message becomes a

group member and removes its own random timer if one is scheduled. The group leader organizes the tracking group, reports tracking data to the base station and initializes leader handover when the object leaves its region. The tracking group organization of EnviroTrack is illustrated in Figure 3.8.

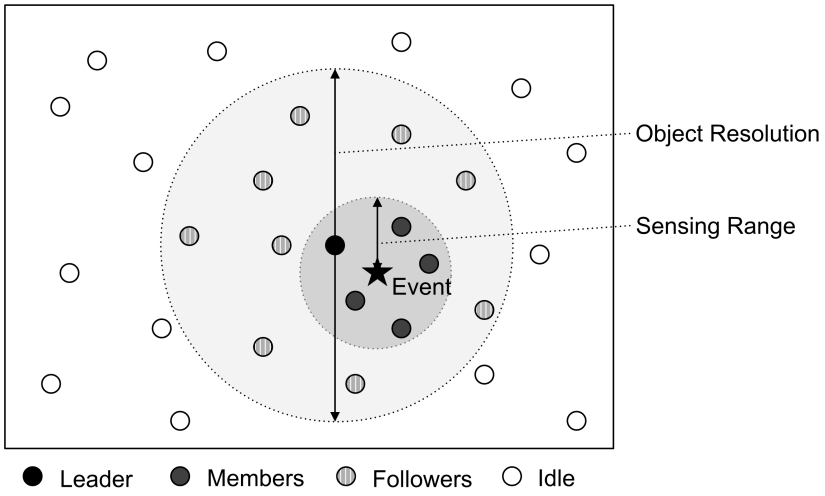



**Figure 3.8:** Group organization in EnviroTrack.

Upon occurrance of an event, every event observing node, i.e., every node in the sensing range, sets a random timer. The node with the fastest exceeding timer (i.e., the black node in Figure 3.8) becomes the leader node. The periodically broadcast heartbeat messages inform all nodes (members and followers) in the transmission range (object resolution area) about the leader. Because EnviroTrack requires that the sensing range is smaller than half the transmission range, the tracking group organization can be maintained very efficiently. This system design imposes low communication overhead, prevents the appearance of concurrent tracking groups and supports efficient leader handover. On the other hand, the sensing range is notably restricted in size. Moreover, the leader node only has local knowledge about the event. This knowledge is however not sufficient to support any event localization or classification. If such functionality is required, collaborative signal processing among the tracking group nodes is needed.

In our own work we have developed DELTA, which performs similar to EnviroTrack, but requires a set of group member nodes to report their sensor readings. Thus, larger sensing areas can be supported, because the report messages cover additional area. Moreover, based on the collected sensor readings, event localization and classification are possible. In DELTA the leader election timer is not set randomly, but based on current sensor readings. Thus, better positioned nodes evolve as leaders. Finally, DELTA implements an on-demand TDMA mechanism to optimize local communication. EnviroTrack has been implemented as the reference algorithm to our own solution. Performance values are presented in Chapter 6.

### 3.4.6 Information-Driven Sensor Querying

In [23] an algorithm for energy-efficient information-driven sensor querying (IDSQ) has been proposed. It is designed to support a wide range of collaborative signal processing tasks such as tracking and classification. The information content, i.e., the sensor readings of interest, is modeled by a probabilistic information utility function. A spatial area of interest is incrementally queried until the given utility function is satisfied. The approach is fully distributed. The management of the event observing area has not been considered, though. Tracking group formation methods such as the ones described before could be used.

In IDSQ, sensor nodes are incrementally queried to provide a dedicated node (the leader node) with the required information. This means, if a group leader node detects an event, i.e., its amplitude readings are greater than a given threshold, it incrementally queries group member nodes until the belief state is considered significant. As soon as the event detection is considered significant, the group leader generates a report message. On the other hand, if all group members have been queried, but the belief function is still not satisfied, the event detection is discarded. This incremental unicast sensor node querying by the leader node is rather time consuming. Therefore, real-time tracking is difficult to support. In addition to the long delays, relying on a unicast communication scheme might be too energy consuming in the context of target tracking and signal processing.

### 3.4.7 Algorithms for Fault-Tolerant Event Region Detection

The detection algorithms discussed so far consider an event significant if some threshold requirements have been met. However, sensor nodes might malfunction and/or (temporarily) provide faulty measurements, which could lead to wrong detection alarms. In the work of [64] the prevention of such false alarms has been addressed.

The authors have developed a distributed Bayesian algorithm which is able to detect and correct wrong measurements caused by malfunctioning sensors. The Bayesian decision procedure requires knowledge of the sensor readings collected in a restricted neighborhood, i.e., in the local event region. The paper does not discuss how the event region is managed by the nodes. This means group formation and maintenance have not been considered in this work. This could however be achieved with one of the previously introduced algorithms. Hence, the tracking algorithms presented in this section could be enhanced with this mechanism to account for faulty measurements.

## 3.5 Event Localization and Signal Strength Estimation

In the previous section we have addressed event detection and event tracking. Having detected an event, it might be necessary to estimate position and/or emitted

signal strength(s) of the event. The computation of the signal strength(s) emitted by the event is, in particular, important for classification.

Accurate event localization and signal strength estimation are commonly addressed by collaborative signal processing (CSP). The focus is thereby on increasing accuracy by multi-measurement processing. High communication and computation costs are often taken into account. The organization of the network into tracking groups is less considered. Some approaches only focus on the solution of localization problems. The state of the art research focuses on statistical and numerical solutions to the localization and signal strength estimation problems. The approaches can be coarsely divided into centralized and distributed approaches. In centralized approaches the sensor information is routed to a central station (base station), which has more storage and computation power than common sensor nodes. The main drawback is high communication load, in particular towards the base station. On the other hand, distributed computation often lacks accuracy and imposes negotiation complexity. In the next section a sensor model that is commonly used to formulate the event-based localization problem is presented. Thereafter, two standard nonlinear optimization methods to solve such problems are introduced. A linear solution to the problem is also presented. The rest of the section discusses work related to event and node localization.

### 3.5.1 Problem Formulation

In order to localize events or to compute their emitted signal strength(s) an appropriate sensor model is needed. Related work [135], [23], [77] uses a sensor model based on an isotropic radiation model (e.g., for sound, vibration, or light from point sources). Thereby, the received signal $\rho_i$ at a sensor node $i$ located at position $\xi_i$ is related to the event position $\mathbf{x}$ according to the model:

$$\rho_i = \frac{c}{\|\mathbf{x} - \xi_i\|^\alpha} + \omega \tag{3.1}$$

where c represents the amplitude of the emitted signal, $\alpha$ is the attenuation degree of the considered signal, $\omega$ is some additional white Gaussian noise, and $\|.\|$ is the Euclidean norm. The model means that the received signal strength decreases inversely proportional to the distance with some exponent $\alpha$. For example, for sound sources $\alpha$ is 2 [63].

Having collected a certain number of sensor readings $\rho_i$, which needs to be larger than the problem dimensionality, the event localization and signal strength estimation problem can be formulated as a nonlinear least-square objective function [114]:

$$f(\mathbf{x}, c) = \sum_{i=1}^{k} \left( \rho_i - \frac{c}{\|\mathbf{x} - \xi_i\|^\alpha} \right)^2 \tag{3.2}$$

Such a function can be solved by nonlinear optimization methods. Alternatively, the system of equations can be linearized and solved (see Section 3.5.4). In some

approaches, the event is not localized based on Equation (3.1), but on other properties such as binary detection information. In this case, Equation (3.2) needs to be adapted. Nevertheless, in order to solve the respective nonlinear function appropriate numerical or statistical techniques are required.

### 3.5.2 Simplex Downhill

First, the Simplex Downhill (SD) method [103] is presented. This method solves a nonlinear optimization problem by searching a minimum in a multidimensional function space. Considering a problem of dimension $N$ with $N \in \mathbb{N}$, a simplex is the simplest volume consisting of $N + 1$ points. If the problem has dimension 2, the simplex is a triangle. Otherwise, if the problem has dimension 3, i.e., there are three unknowns, the simplex is a tetrahedron. We show a simple example of a nonlinear function minimization problem in Figure 3.9. This problem could be solved analytically, but illustrates the problem well.

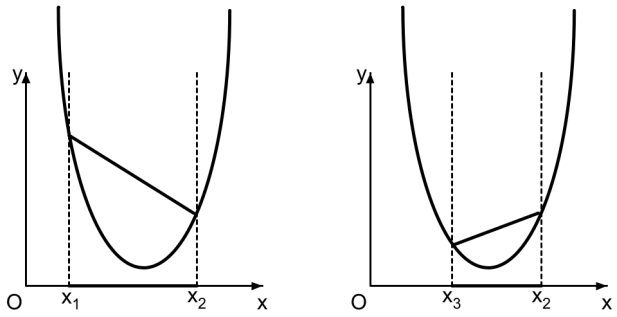



**Figure 3.9:** Simplex Downhill function minimization.

In this example, the problem dimensionality is 1. Accordingly, there is only one unknown variable $x$ and the simplex is a line. Initially, a well-placed simplex is chosen, i.e., the line $\overline{x_1 x_2}$ in Figure 3.9. Then, the objective function is applied on it. The mapped simplex $f(\overline{x_1 x_2})$ is then processed with a sequence of simple geometrical operations such that the minimum of the objective function is searched. To do so, the highest value in the simplex is always chosen and transformed to become the smallest one. In Figure 3.9 f($x_1$) initially has the highest value. Accordingly, it is transformed to become the smallest one (f($x_3$)). A sequence of transformations is repeated until either a given threshold is under-run or the maximum number of allowed iterations is reached. The possible geometrical operations of a simplex (tetrahedron in this case) are depicted in Figure 3.10.

The SD method requires no derivations, but only simple function evaluations. On the other hand, the convergence to the searched minimum needs on average more steps than with methods that consider derivations. A local minimum could be found that depends of the location of the starting simplex. Therefore, a well located starting simplex is crucial. Up to now only heuristics exist to avoid local minima,

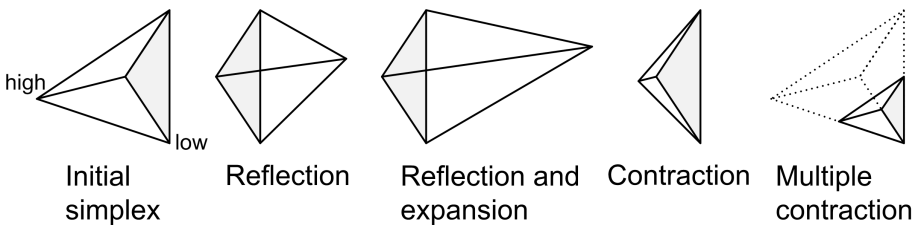



**Figure 3.10:** Geometrical operations of Simplex Downhill.

e.g., Monte Carlo methods, which address this problem. Such methods are very cost intensive in terms of time and memory, though. Therefore, they are in general not applicable to sensor networks.

### 3.5.3 Conjugate Gradient Method

The Conjugate Gradient [114] method also solves nonlinear optimization problems of dimension N. At a given N-dimensional point **P**, not only are the function evaluations $f(\mathbf{P})$ computed, but the gradients of the objective function $\Delta f(\mathbf{P})$ as well. The gradient $\Delta f(\mathbf{P})$ is a vector field that has the characteristic of pointing in the direction of the largest increase of $f(\mathbf{P})$. In its simplest form this optimization problem can be solved by searching in the negative direction $-\Delta f(\mathbf{P})$. This method is called Steepest Descent. An example with both Conjugate Gradient and Steepest Descent search is illustrated in Figure 3.11.

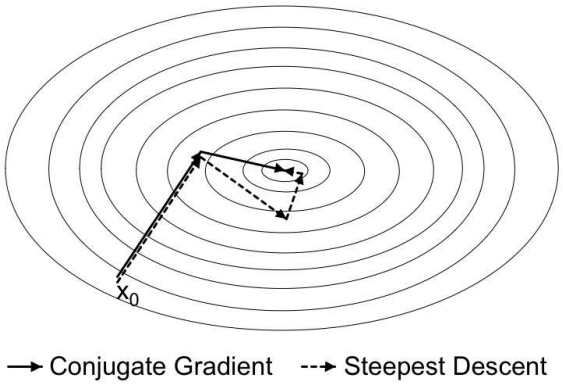



**Figure 3.11:** Conjugate Gradient and Steepest Descent.

In the example, the objective function is a three-dimensional cone. The function minimum is indicated by the decreasing ellipses. A starting point $x_0$ is determined. At this point, the search starts in the negative direction of the gradient, i.e., in the direction of the arrows in Figure 3.11. As soon as a minimum in that direction is reached, the search direction is adapted such that it again points in the negative direction of the gradient (Steepest Descent). With CG this direction is slightly

adjusted such that in general fewer iterations are needed to reach the minimum (2 steps in the example). Commonly, the Steepest Descent method needs more steps to terminate (4 steps in the example). Like SD, CG can enter a region which ends in a local minimum. Therefore, CG faces the same restrictions as SD. The algorithm again terminates if the termination criterion is satisfied or if the maximum number of allowed iterations is reached.

### 3.5.4 Linear Least Square Method

In the previous two sections two nonlinear methods to solve source-based localization and signal strength estimation problems have been presented. The problem can further be linearized and solved with Linear Least Square (LLS) methods [135], [78], [77]. In [135], [77] acoustic event sources have been localized. The signal attenuation coefficient $\alpha$ of acoustic sources is 2 (see Section 3.5.1). Accordingly, Equation (3.1) can be rewritten as:

$$\|\mathbf{x}\|^2 + \|\xi_i\|^2 - 2\mathbf{x}^T\xi_i - \frac{c}{\rho_i} = 0 \tag{3.3}$$

Given N sensors, N equations (3.3) can be formulated. The quadratic constraints on the unknown variable $\mathbf{x}$ can be removed by subtracting the first (i = 1) equation from the rest (i $\neq$ 1), resulting in a system of N-1 linear equations of the form

$$2\left(\xi_1 - \xi_i\right)^T \mathbf{x} + c\left(\frac{1}{\rho_1} - \frac{1}{\rho_i}\right) = \|\xi_1\|^2 + \|\xi_i\|^2 \tag{3.4}$$

In the following the unknown variables are rearranged in a vector $\hat{\mathbf{x}} = [\mathbf{x}, c]$. By setting

$$a_i = \left[2\left(\xi_1 - \xi_i\right); \left(\frac{1}{\rho_1} - \frac{1}{\rho_i}\right)\right]; \qquad b_i = \|\xi_1\|^2 + \|\xi_i\|^2$$

Equation (3.4) can be simplified to $a_i^T\hat{\mathbf{x}} = b_i$. Considering all N-1 linear constraints, the system can be written in matrix form as $\mathbf{A}_{N-1}\hat{\mathbf{x}} = \mathbf{b}_{N-1}$, which can be solved with the following closed-form standard linear least square method $E = (\mathbf{A}_{N-1}^T\mathbf{A}_{N-1})^{-1}\mathbf{A}_{N-1}^T\mathbf{b}_{N-1}$. In order to apply the linear least square method, an over-determined system is required. This means the number of sensing nodes needs to be larger than n + 1, where n is the problem dimensionality. In sensor networks this requirement can be restrictive.

### 3.5.5 Tracking a Moving Object with a Binary Sensor Network

A centralized approach for object tracking has been proposed in [4]. Based on a binary sensor model, the location and direction of a moving object are estimated. The only information a binary model requires is whether an object is moving towards a sensor node or away from a sensor node. The binary information is collected at a central station where a particle filter algorithm is applied to estimate location and

movement direction of the object. A particle filter is a statistical estimation method that is based on sequential Monte Carlo simulations. In addition to the currently observed measurements, the algorithm also requires previous information (some history) about the moving object.

The authors show that the binary model is appropriate if only direction information about the object is required. However, if position information is needed, an additional sensor measurement which indicates proximity is required. Another limitation of the algorithm is its centralized nature. In a distributed implementation the particle filter might be too resource consuming due to the simulation-based estimation. To some extent the classification of objects according to their velocity might be possible. If however information such as emitted signal strength estimations of the objects is needed, the approach is not applicable, because of its dependency on only binary information.

### 3.5.6 PinPtr: Sensor Network-Based Countersniper System

In [134] the authors have proposed a centralized sniper detection system where each event observing node estimates the distance to a sniper based on the time difference of arrival (TDOA) between two different kinds of signals. The TDOA between the arrival of a muzzle blast and an acoustic shockwave is measured. These estimates are delivered to a base station, where the position of the sniper is computed by a multidimensional sensor fusion algorithm. The unknown sniper position and the measurements span a four-dimensional vector space, which is searched for the maximal set of consistent measurements by performing a Generalized Bisection method [54]. This is a reliable nonlinear optimization method that divides the solution space into subregions. The subregions are searched by the Newton-Raphson method [114]. The approach is again very resource consuming and accordingly performed at a dedicated base station.

To determine the positions of the sensor nodes an additional self-localization service is provided. This service performs pair-wise ranging based on acoustic and radio signals. All ranging measurements are transmitted to a base station, where an optimization procedure is performed, which iteratively places the nodes relative to some known anchor nodes.

Exact time synchronization is required for both, sensor node positioning and sniper localization. The authors state that the communication burden of transmitting the TDOA measurements of all sensing nodes to a base station is acceptable for their application. In [146] the system has been adapted to operate in mobile environments. In addition to sniper localization, the classification of bullet calibers and specific weapons is supported. The bullet caliber classification is based on the relation of the shockwave period to the bullet characteristics and the missing distance between the bullet trajectory and a sensing node. The weapon classification is based on the projectile speed and its caliber.

### 3.5.7 Localization with Positive and Negative Constraints

Distributed coarse-grained node and source localization has been proposed in [41], [44]. No classification of the localized sources is intended. Sextant [44] uses Bézier regions to represent the locations of both nodes and event sources. The regions are constructed by gathering positive and negative connectivity constraints in the neighborhood. To achieve this, Sextant nodes disseminate their monitored network properties in a restricted area for a predefined number of hops. Drawbacks of the algorithm are rather high delays and limited localization accuracy. Classification might be possible if, in addition to connectivity information, the sensor readings were disseminated too. The authors of [41] have proposed a similar approach, but have used rectangles instead of Bézier regions.

### 3.5.8 SensIt: Distributed Localization and Signal Processing

Source localization based on the sensor model presented in Section 3.5.1 has been investigated in the SensIt project [78], [76], [77]. In contrast to PinPtr, which needs two distinct signals for range estimations, the algorithms developed in the SensIt project require only one specific signal, e.g., acoustic or seismic.

In [78] the focus has been on the localization and classification of sources based on their seismic fingerprint. A seismic signal is modeled according to Equation (3.1). Event detection and tracking are performed within space-time regions (cells) as described in Section 3.4.2. The tracking group leader(s) collect(s) time series of seismic measurements. The localization of single targets is performed with over-determined linearized least-square (LLS) methods (see Section 3.5.4). In subsequent research [77] acoustic localization has been performed instead of seismic localization. Again a closed-form linearized least-square method has been used. As mentioned before, more data is required, but considering real-time performance, a closed-form solution to the localization problem appears very attractive. Closed-form solutions for acoustic localization had been proposed before, e.g., in [135]. Apart from source localization, linearized least-square methods have been widely used in range-based node positioning, e.g., in [125].

The LLS method requires an overdetermined system to work accurately. To overcome this restriction, alternative nonlinear numerical optimization methods have been evaluated in [76]. Again, the position of an event radiating signals is estimated based on collected sensor readings. Two simple brute-force methods, i.e., Exhaustive Search (ES) and Multi-resolution search (MR), have been evaluated together with the Simplex Downhill algorithm (SD) (see Section 3.5.2) and the Conjugate Gradient (CG) descent method (see Section 3.5.3). In order to minimize the risk of finding a local optimum, the feasible solution space has been overlaid by a grid. The respective optimization procedure is performed at every point in this grid. The global maximum is chosen as the maximum among all results. Overlaying the whole solution space is too complex to be considered in a distributed sensor network application.

In our own work we use SD and CG too (see Chapter 7). However, we determine well located starting points to avoid local optimums. Our method has the advantage, that meaningful localization results can be achieved even with the minimum amount of required data. In wireless sensor networks redundant sensor data might often not be available. Moreover, collecting redundant sensor readings imposes communication costs.

### 3.5.9 Robust Localization of Multiple Events

In [19] the localization of single events as well as of multiple coexisting events has been investigated. Again, the omnidirectional energy-decay model has been used. The authors show that linearized methods fail in specific network topologies. In particular if the sensor nodes are arranged on a line.

The position of single events is computed with two probabilistic models, namely a Minimum Mean Square Error (MMSE) estimator and a Maximum A Posterior (MAP) estimator. The calculation of the estimates is computationally expensive. Therefore, a voting-based approximation procedure is used. To determine the locations of multiple coexistent sources, the authors propose to use a nonlinear optimization method. A Levenberg-Marquart algorithm [75] that locally performs the Newton-Raphson method [114] has been used. All proposed algorithms are resource consuming and require high communication load. Networking issues, such as how to establish and maintain tracking groups, have not been addressed.

### 3.5.10 Distributed Optimization in Sensor Networks

The authors of [118] investigate the feasibility of deriving an estimation distributed in the network. This avoids the need of sending all data to a central node, where the computation would be performed. The method resembles the IDSQ sensor querying procedure presented in Section 3.4.6. The event-based localization of acoustic sources is investigated as a possible application. The localization problem is expressed as an optimization problem, whereby the nonlinear problem (cost) function is incrementally fed with new measurements. Additional measurements are obtained by circulating the cost function in the network. Every receiver of the cost function updates the function with its own local sensor readings, i.e., with its local acoustic signal strength measurement, until a precision threshold is reached, or until the maximum number of allowed search steps is exceeded. In each step an incremental subgradient optimization method, processing the local data, is applied.

The approach is suitable for estimations problems such as the localization or classification of static sources. However, if moving sources are present, the parameter circulation paradigm counteracts the tracking task, which requires some kind of tracking group formation. The node terminating the optimization could be designated responsible for the tracking. Nevertheless, the circulation paradigm implies delays, which might negatively affect real-time performance in a dynamic event monitoring context.

### 3.5.11 Distributed State Representation for Tracking Problems

In [87] a tracking problem is addressed by decomposing the tracking problem into a positioning problem (space) and into an identity problem (state). The state-space model embodies such a problem. The approach aims at distinguishing mutually overlapping events. The basic idea is to decompose the joint state-space of any event into sub-problems of lower dimensionality. Considering tracking, the joint state-space covers both positioning and identity management. This joint problem is decoupled into two sub-problems, namely into identity management and target positioning. Both sub-problems are of lower complexity than the joint problem. Each of the sub-problems is processed locally on the sensor nodes of interest. The sensor nodes of interest are dynamically determined according to application-specific state-space requirements.

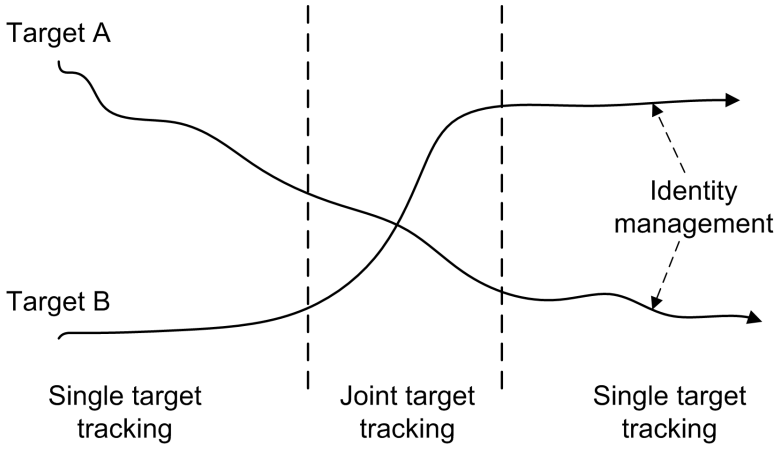



**Figure 3.12:** Decomposition of multi-target tracking.

The decomposition of the tracking problem into localization and identity management sub-problems is depicted in Figure 3.12. As long as two targets are disjointed, the positioning and identity management sub-problems are addressed individually for both targets. As soon as the target tracking areas overlap, the identity management problem is abandoned due to complexity. Instead, both targets are treated as a single target and only joint positioning is performed. As soon as the targets are sufficiently disjointed in space again, the sub-problems are again addressed individually. The localization and identity check procedures are based on statistical methods and require knowledge about the history of sensor measurements. Thus, they are rather expensive in terms of storage and communication.

### 3.5.12 Classical Node Positioning Methods

In the following a number of classical node positioning approaches are discussed. These methods have not yet been adapted to source localization and/or classification problems, but could be of some relevance in that respect too. Good overviews

over early work in this topic are provided by [73], [105] and [52]. These papers discuss approaches such as DV-Hop [106], Cricket [115], and so on, which are interesting but less relevant in our context.

### Multidimensional Scaling (MDS)

In [130] a coarse-grained node localization method, only requiring connectivity information, based on multidimensional scaling has been proposed. A relative map containing the distances (in hops) between the network nodes is generated. On this map multidimensional scaling is applied to derive the node positions which best fit the distance estimates. The resulting positions are relative to the map. In order to obtain absolute coordinates, landmarks with known position can be used to normalize and transform the relative coordinate system according to trigonometric properties. Instead of distance other properties could be used. The high signaling burden, i.e., all data needs to be collected at a central node, and the high computational requirements make the approach less suitable for tracking and classification in sensor networks. In a subsequent paper [129], the approach has been distributed by decomposing the global map into local maps. The local maps can be combined if required. A distributed variant based on Received Signal Strength Indicator (RSSI) measurements has been proposed in [58]. Due to the usage of RSSI this algorithm provides a more fine-grained resolution.

### Multilateration for Node Localizations over Multiple Hops

In [126] the multilateration problem is split into two sub-problems. Coarse-grained location information is collected from landmarks which are placed several hops away. Additionally, neighboring nodes measure their mutual distances based on ultrasonic ranging. In a first step, nodes are organized into groups such that nodes with unknown position are over-constraint and a unique solution can be derived. Then, coarse-grained initial position estimates are obtained from simple geometric relationships. Finally, a distributed nonlinear gradient descent method is applied to derive the final fine-grained location estimates. Due to its communication requirements this method is only appropriate if infrequently performed. In previous work [125], [73] standalone multilateration had been used.

### Localization Based On a Kernel Method

In the work of [104] the localization problem is formulated as a pattern recognition problem. A signal strength matrix containing the mutual measurements of each pair of sensor nodes is required. Furthermore, some landmarks are needed. The classification problem is solved by using support vector machine (SVM) and kernel methods. The localization is divided into a training phase, where the localization functions are learned from the signal strength matrix with respect to the landmarks. This is performed centrally at a base station. Afterwards, each sensor node with unknown location performs its positioning locally. The algorithm does not require

distance estimations. A drawback is that the preprocessing needs to be performed at a central base station. Moreover, the algorithm requires a high communication load if the preprocessing has to be performed frequently, which would be the case in a source tracking and localization application. Similar localization procedures based on kernel methods have been proposed in [178] and [66].

### Localization Considering Non-Line-Of-Sight Measurements

In [145] the node localization problem is addressed in environments which consider both, line-of-sight (LOS) and non-line-of-sight (NLOS) measurements. The resulting system is solved with linear programming. The NLOS estimates are considered if the LOS measurements cannot guarantee accurate localization. The LOS measurements are used to define the objective function, while the NLOS measurements add restrictions to the feasible solution space for the linear program. Mechanisms to deal with NLOS measurements might be of interest in source localization and classification too. The system must be able to distinguish between LOS and NLOS measurements.

## 3.6 Classification and Reasoning

The deployment of wireless environmental monitoring systems that include classification primitives is still a challenging problem. First of all, classification software consumes a lot of resources in terms of storage, processing and communication. Furthermore, sensor networks commonly aim at avoiding resource consuming mechanisms. Thus, there is a trade-off between efficient distributed classification and the available resources provided by sensor networks. Reporting all data to a base station for later data analysis (e.g., [5]) is too communication intensive in our context. Such a system design is only appropriate for short-term deployments. On the other hand, our event detection system aims at long-term deployments. Therefore, lightweight and efficient, distributed classification mechanisms are needed.

Before discussing state of the art classification methods in wireless sensor networks, some basic classification aspects are introduced. Particular focus is on classification aspects and features that are relevant for our own approach.

Basically, a classification problem is the problem of assigning a present unknown pattern $\mathbf{x} = \{x_1, ..., x_N\}$ to the class $C_i$ of known patterns it most likely belongs to. $N$ is the number of features that characterize the pattern.

In our work the event classes are learned unsupervised, i.e., they are learned from observed data. A well-known algorithm to learn event classes is presented in the next section. The subsequent three sections introduce three different basic classification methods, namely a simple Bayesian classifier, a classifier based on Fuzzy Logic, and a neural network approach. These methods aim at classifying patterns that are present as discrete entities in time, i.e., the input pattern is of form $\mathbf{x}$. In addition, the theory of ART neural networks is presented. These kinds of neural networks are used in our own work to process and classify events that evolve

over time. Having presented these mechanisms, the focus is placed on related work in classification and anomaly detection in wireless sensor networks.

### 3.6.1 Learning Event Classes

Event classes are learned from training data. We will use two known clustering mechanisms, namely k-means and fuzzy k-means. Both mechanisms arrange sample patterns into clusters. Therefore, similarity metrics, i.e., the Euclidean distance (k-means) and membership degrees (fuzzy k-means), are used.

The goal of clustering is to find a decomposition of the (training) set $\mathbf{Z} = \{\mathbf{z}_1, ..., \mathbf{z}_M\}$ into $m$ clusters $\{C_1, ..., C_m\}$. The basic idea of fuzzy k-means is to assign each $\mathbf{z} \in \mathbf{Z}$ to each cluster $C_j$ with a given membership degree $\mu_j(\mathbf{z})$. Hard clustering (k-means) is a special case of fuzzy clustering with $\mu_j(\mathbf{z}) \in \{0, 1\}$ for all j. The fuzzy k-means algorithm requires the computation of the membership degree of a sample pattern $\mathbf{z}$ to a cluster $C_j$ according to [39]:

$$\mu_j(\mathbf{z}) = \begin{cases} 1, & \text{if } \mathbf{z} = \mathbf{m}_j, \\ \frac{1}{\sum_{k=1}^{K} \left( \frac{\|\mathbf{z}-\mathbf{m}_j\|}{\|\mathbf{z}-\mathbf{m}_k\|} \right)^{\frac{2}{\beta-1}}}, & \text{else.} \end{cases} \tag{3.5}$$

where $\beta$ is a parameter controlling the membership gradient and $\|.\|$ is the Euclidean norm. According to (3.5) the membership degree of a sample pattern $\mathbf{z}$ is higher, the closer $\mathbf{z}$ and a cluster center $\mathbf{m}_j$ are. The computation of the cluster centers $\mathbf{m}_j$ is based on the membership degrees of all patterns $\mathbf{z} \in \mathbf{Z}$ [39]:

$$\mathbf{m}_j = \frac{\sum_{i=1}^{M} \mu_j(\mathbf{z}_i) \cdot \mathbf{z}_i}{\sum_{i=1}^{M} \mu_j(\mathbf{z}_i)} \tag{3.6}$$

Having defined these preliminaries, the k-means and the fuzzy k-means algorithms can be written as:

**K-means**

**input:** Training set $\mathbf{Z}$;
$K =$ number of clusters;
**ouput:** $m$ clusters $\{C_1, ..., C_m\}$;
**begin**
choose $K$ initial cluster centers $m_1, ..., m_K$;
**repeat**
assign each $\mathbf{z}_i$ to the cluster with closest center $\mathbf{m}_j$;
recompute each cluster center $\mathbf{m}_j$;
**until** termination criteria satisfied;
**end**

**Fuzzy k-means**

**input:** Training set $\mathbf{Z}$;
$K =$ number of clusters;
**ouput:** $m$ clusters $\{C_1, ..., C_m\}$;
$\mu_j(\mathbf{z}_i)$ for $1 \leq j \leq K$ and $1 \leq i \leq M$;
**begin**
choose $K$ initial cluster centers $m_1, ..., m_K$;
**repeat**
compute $\mu_j(\mathbf{z}_i)$ for $1 \leq j \leq K$ and $1 \leq i \leq M$ according to (3.5);
update the cluster centers $\mathbf{m}_j$ for $1 \leq j \leq K$ according to (3.6);
**until** termination criteria satisfied;
**end**

The fuzzy k-means algorithm works similar to the k-means algorithm. The main difference is that it does not assign a fixed pattern to a cluster, but assigns a pattern to a cluster with a certain membership degree. This has also some impact on the computation of the cluster centers, because the impact of all samples is considered instead of only the samples that belong to the specific cluster.

Having applied any of the two clustering algorithms on a training set, the resulting clusters (classes) can be used to configure the classifiers discussed in the next section. Both clustering algorithms can be used as substitutes because both mechanisms produce similar clusters.

### 3.6.2 Bayesian Classifier

It is assumed that $m$ different event classes (clusters) $C_i$ have been learned. Each pattern $\mathbf{x}$ of $C_i$ is an element of $\Re^n$, where $n$ is the total number of observed phenomena (features).

To implement a Bayesian classifier, first the a priori class probabilities $p(C_i)$ need to be known. These probabilities represent the frequency of every class $C_i$ over all known patterns. Moreover, the class-specific probabilities $p(\mathbf{x}|C_i)$ are required. $p(\mathbf{x}|C_i)$ is the probability to which extend $\mathbf{x}$ belongs to $C_i$. A Bayesian classifier implements the following classification rule [72], [39]:

$$\mathbf{x} \in C_i \Leftrightarrow p(\mathbf{x}|C_i)p(C_i) > p(\mathbf{x}|C_j)p(C_j), \quad \forall j = 1, ..., m; j \neq i$$

A Bayesian classifier assumes normal distributions of the probabilities $p(\mathbf{x}|C_i)$. Normal distributions are analytically easily manageable. The parameters, i.e., the mean value $\mathbf{m}$ and the standard deviations $\mathbf{K}$ of each cluster, can easily be estimated from the patterns in the different clusters $C_i$. The normal distribution in $n$ dimensions, with $n \geq 2$, looks as follows [39]:

$$p(\mathbf{x}|C_i) = \frac{1}{(2\pi)^{\frac{n}{2}}|\mathbf{K}_i|^{\frac{1}{2}}} e^{\left[-\frac{1}{2}(\mathbf{x}-\mathbf{m}_i)'\mathbf{K}_i{}^{-1}(\mathbf{x}-\mathbf{m}_i)\right]} \tag{3.7}$$

This function only requires the knowledge of the mean vectors $\mathbf{m}_i$ and the covariance matrices $\mathbf{K}_i$ for each class $C_i$. The covariance matrix is a $n \times n$ matrix. $|\mathbf{K}_i|$ is the determinant of $\mathbf{K}_i$. Instead of using $p(\mathbf{x}|C_i)p(C_i)$ as a classification rule (3.7), a monotone function can be applied to simplify the computation. As done by other authors [72], [39], a natural algorithm is used:

$$D_i(\mathbf{x}) = ln[p(\mathbf{x}|C_i)p(C_i)] \tag{3.8}$$

Thus, the new classification rule is:

$$\mathbf{x} \in C_i \Leftrightarrow D_i(\mathbf{x}) > D_j(\mathbf{x}), \quad \forall j = 1, ..., m; j \neq i \tag{3.9}$$

Substituting (3.7) in (3.8) and ignoring the emerging constant term $\frac{n}{2}ln(2\pi)$ the computation of $D_i$ looks as follows:

$$D_i(\mathbf{x}) = ln[p(C_i)] - \frac{1}{2}ln|\mathbf{K}_i| - \frac{1}{2}(\mathbf{x} - \mathbf{m}_i)'\mathbf{K}_i{}^{-1}(\mathbf{x} - \mathbf{m}_i) \tag{3.10}$$

The Bayesian classifier is fully functional as soon as the the covariance matrix $\mathbf{K}_i$ and the mean $\mathbf{m}_i$ of each cluster $C_i$ have been computed. Both can be computed in a straight forward manner based on the cluster information provided by the k-means algorithm. Having patterns $\{\mathbf{x}_1, ..., \mathbf{x}_M\}$ of a class $C_i$, the parameters $\mathbf{K}_i$ and $\mathbf{m}_i$ are estimated as follows [39]:

$$\mathbf{m_i} = \frac{1}{M}\sum_{j=1}^{M}\mathbf{x}_j, \qquad and \qquad \mathbf{K}_i = \sum_{j=1}^{M}\mathbf{x}_j\mathbf{x}_j' - \mathbf{m}\mathbf{m}'$$

A Bayesian classifier is a simple classifier that assumes normal distribution of the features in the patterns. If this is not the case, Bayesian classifiers intrinsically lose precision due to their mismatching distribution assumption.

### 3.6.3 Fuzzy Logic Controller

In this section the basic design of a common Fuzzy Logic Controller (FLC) is discussed according to [72]. We assume that $m$ different event classes $C_i$ have been learned (see Section 3.6.1). Let $U = \{u_1, ..., u_n\}$ be the universal set. A fuzzy set $\tilde{A}$ on $U$ is described by the membership function:

$$\mu_{\tilde{A}} : U \rightarrow [0, 1] \tag{3.11}$$

where $\mu_{\tilde{A}}(u)$ expresses the membership degree in which the element $u$ belongs in $\tilde{A}$.

In our work, we have investigated triangular and Gaussian membership functions. The functions are depicted in Figure 3.13 and computed according to:

$$\mu(x) = \begin{cases} \frac{x-a}{b-a}, & \text{if x} \in (a, b], \\ \frac{c-x}{c-b}, & \text{if x} \in (b, c], \\ 0, & \text{otherwise.} \end{cases} \quad and \quad \mu(x) = e^{-\frac{(x-a)^2}{2\sigma^2}}$$

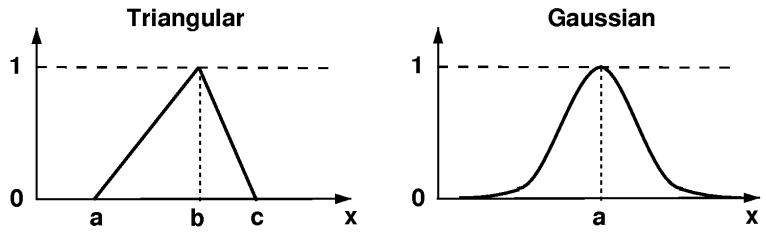



**Figure 3.13:** Membership functions used in fuzzy logics.

Gaussian functions have the general advantage that the whole feature space is covered. Considering a classification problem with $\mathbf{x} \in \Re^n$ and $m$ event classes, a fuzzy logic controller consists of a system of $m$ fuzzy if-then rules $R_k$ of the form [72]:

$$\begin{aligned} R_k : IF\ \ & \mu_{\tilde{A}_{k,1}}(x_1) \wedge \mu_{\tilde{A}_{k,2}}(x_2) \wedge ... \wedge \mu_{\tilde{A}_{k,n}}(x_n) \\ & THEN\ \ g_k(\mathbf{x}) > g_i(\mathbf{x}),\ \ \forall i = 1, ..., m \end{aligned} \tag{3.12}$$

where $\wedge$ is an arbitrary operator aggregating the fuzzy sets of the premises and $g(\mathbf{x})$ is an arbitrary function of the consequence. The rule means that the degree of significance of the premise is assigned to the conclusion. If the premise of $R_k$ is fulfilled to a high degree, then the consequence of $R_k$ has a high degree of significance also. Considering a sample pattern $\mathbf{x}$, the degree of significance of rule $R_k$ should be maximum if $\mathbf{x}$ belongs to class $C_k$.

The function $g_k(\mathbf{x})$ of the consequence can be an arbitrary function if the FLC is designed as a Takagi-Sugeno classifier [72]. Takagi-Sugeno classifiers have the advantage that their parameters can be estimated from training data. The premises are modeled from the patterns in the clusters $C_i$. Any TSK classifier implements rules of the general form (3.12) in the following way [72]:

$$\begin{aligned} R_k : IF\ \ & \mu_{\tilde{A}_{k,1}}(x_1) \wedge \mu_{\tilde{A}_{k,2}}(x_2) \wedge ... \wedge \mu_{\tilde{A}_{k,n}}(x_n) \\ & THEN\ \ g_k(\mathbf{x}) = f_k(\mathbf{x}) \end{aligned} \tag{3.13}$$

where $f_k(\mathbf{x})$ is an arbitrary function. Because the functions $f_k(\mathbf{x})$ can be chosen arbitrarily, they can be designed to model the learned event classes $C_i$ based on the patterns in the $C_i$. Accordingly, no expert knowledge is required. In our work we have used a TSK2 classifier that is specified as follows [72]:

**TSK2 classifier**

- $z_{k,i} \in \Re,\ \ k = 1, ..., m,\ \ i = 1, ..., m$;
- The conjunction (AND) is the product;
- The $i$th output of TSK2 is

$$g_i^{TSK2}(\mathbf{x}) = \frac{\sum_{k=1}^{M} z_{k,i} \prod_{j=1}^{n} \mu_{\tilde{A}_{k,j}}(\mathbf{x}_j)}{\sum_{k=1}^{M} \prod_{j=1}^{n} \mu_{\tilde{A}_{k,j}}(\mathbf{x}_j)} \tag{3.14}$$

After having determined the fuzzy sets of the premises, the fuzzy consequences $g_i(\mathbf{x})$ can be tuned for the TSK2 classifier (see [72], p. 182). This means that estimates of the $z_{k,i}$ have to be computed.

Even though TSK classifiers can be modeled from data and are therefore trained unsupervised, most current applications implement the consequences of the rules according to Mamdani inference [95]. These systems require expert knowledge and are based on minimum/maximum decisions. In particular, these systems require that the consequences of the classification rules are linguistic variables. This restricts the application of the system, because linguistic variables are often difficult to model and require expert knowledge as mentioned before.

### 3.6.4 Feedforward Neural Networks

Feedforward Neural networks (FFNN) [39] are bio-inspired networks which, in general, are neither self-documenting nor directly comprehensible for human beings. They consist of simple, mutually connected computing units (neurons), which support parallel computing, learning, and generalization.

Basically, FFNNs consist of an input layer, an output layer, and one or more hidden layers of neurons. The number of input variables, output variables and neurons is arbitrary. Again, input is the unknown pattern $\mathbf{x} = \{x_1, ..., x_N\}$ that contains the different event characterizing features. Output is the class $C$ to which the pattern most likely belongs. The hidden layer(s) consists of neurons that process the input $\mathbf{x}$ according to some weights $w_i$. A simple neuron is defined as follows:

$$f(x_1, ..., x_n) = \sum_{i=1}^{n} w_i x_i \tag{3.15}$$

A weight $w_i$ is assigned to each input variable $x_i$. In FFNNs these weights are learned from training data consisting of input patterns $\mathbf{x}$ and the associated output, i.e., the class $C_i$ to which $\mathbf{x}$ belongs. The training is based on the backpropagation method (e.g., [39]), which feedforwards the input training pattern throughout the neural network, analyzes the error by backpropagation (nonlinear optimization), and updates the weights. The backpropagation procedure is based on the Steepest Descent method, which performs similar to the Conjugate Gradient method (see

Section 3.5.3). Once the weights have been learned from training patterns, the FFNN operates efficiently since only simple arithmetic operations are performed. The determination of the weights can be very resource consuming, though. In our own approach we have used a FFNN method too. The weights are determined offline at a base station. Only the trained, efficient classifier is downloaded onto the sensor nodes. This is the same for the FLC and the Bayesian classifier.

### 3.6.5 Adaptive Resonance Theory

If only little knowledge of the expected kinds of events is available, the previously proposed algorithms fail due to their limited online learning capability. In this section we introduce a lightweight and adaptive memory approach that learns and classifies event patterns online. Adaptive Resonance Theory (ART) neural networks [17] represent a special kind of adaptive memory with sequential learning ability. Any present input $\mathbf{x} = \{x_1, ..., x_N\}$ is fed into the ART neural network. The present input is classified with respect to a number of stored prototypes, which represent learned classes of input patterns $\mathbf{x}$. If the present input can be classified, the respective prototype is updated. Otherwise, a new prototype is created unless the whole memory capacity is utilized. ART systems have been designed to process binary input patterns (binary ART) and analog input patterns (Fuzzy ART).

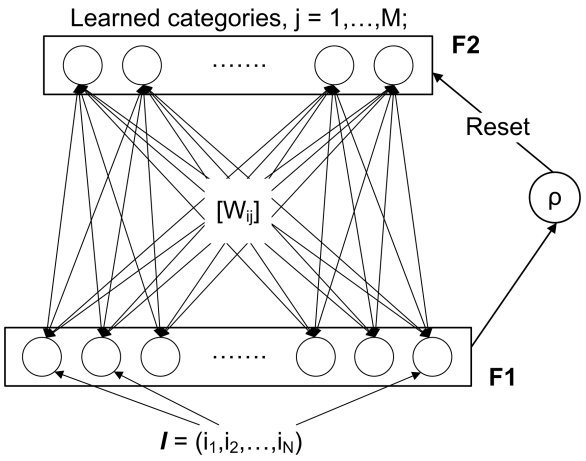



**Figure 3.14:** ART neural network architecture.

The architecture of any ART neural network is shown in Figure 3.14. Any ART neural network is an unsupervised learning system. It consists of two layers, a comparison layer F1 with $N$ neurons representing the attributes of a given input and a recognition layer F2 composed of $M$ neurons representing the prototypes (categories). The weight matrix $W_{i,j}$ is the memory of the ART. The sensitivity threshold $\rho$ controls the recognition behavior of the ART neural network.

**ART-based Event Recognition**

**input:** Input vector $\vec{I}$;
**output:** Number representing category $j$ to which $\vec{I}$ belongs;
**begin**
    Compute similarity $s_j$ to each prototype j in F2;
    Sort the $s_j$ in descending order;
    **for each** $s_j$ **do**
        **if** $s_j > \rho$
            Update the weights $W_{:,j} = \vec{I} \cdot \alpha + W_{:,j} \cdot (1 - \alpha)$ ;
            **return** Category number j;
    **if** maximum number of categories is not reached
        Commit uncommitted neuron $n$ in F2;
        **return** Category number j;
    **else**
        replace oldest category in F2;
        **return** -1;
**end**

The operation of an ART-based event recognizer is described in the pseudo-code above. The weight matrix $W_{i,j}$ is the memory of the ART. First, the similarity $s_j$ between every prototype $j$ and the input vector $\vec{I}$ is determined. The Euclidean distance is used to determine similarity $s_j$ between $\vec{I}$ and any of the stored prototypes $j$. The resulting list of similarities is sorted in descending order. The ordering is necessary, otherwise a prototype might be chosen (similarity applies), though a more similar prototype exists. This case could happen if more than one $s_j$ exceeds the sensitivity threshold $\rho$. The resulting list of similarities is evaluated with respect to $\rho$. If an appropriate category is found, the weights of the according prototype are updated and the category number is returned as classification output. If no category could be determined, -1 is reported (unknown event pattern). The present input $\vec{I}$ is stored as new prototype until the memory is full.

The parameters of the ART neural network are explained in the following. If $s_j$ exceeds sensitivity threshold $\rho$, $\vec{I}$ is assigned to category j. A high value for $\rho$ implies fine-grained memory (many, small categories), since the input needs to match a category exactly. On the other hand, low values mean coarse recognition (few, large categories). A second parameter that has an impact on the behavior of the ART neural network is the learning rate $\alpha$. If an input $\vec{I}$ is assigned to a category $j$, the stored prototype $j$ is updated according to the weighted sum of $\vec{I}$ and $j$, i.e., $W_{:,j} = \vec{I} \cdot \alpha + W_{:,j} \cdot (1 - \alpha)$. Hence, the learning rate defines the weights $\alpha$ for the input and $(1 - \alpha)$ for the stored prototype. If $\alpha$ is high (e.g., 0.8), $\vec{I}$ is weighted 0.8 and the stored prototype $j$ is weighted 0.2. Accordingly, high learning rates reinforce the impact of the current input. Traditional ART neural networks return the category number if a category is determined for a given input $\vec{I}$ and -1 otherwise. On the other hand, our ART-based event recognizers return 0 (known) if $\vec{I}$ is recognized and 1 (unknown) otherwise.

ART neural networks are lightweight and adaptive. No buffering of event patterns is needed. Only a small number of prototypes is maintained. However, because event classes are modeled according to weighted prototypes, the mechanism cannot achieve the accuracy of an algorithm that stores all observed event patterns. We use an adapted ART neural network mechanism for anomaly detection. Moreover, we extend common ART neural networks with an aging mechanism. Thus, learning capability can be continuously maintained.

### 3.6.6 Haar Wavelet Transform

On wireless sensor nodes memory capacity is critical. Therefore, the size $N$ of the input vector $\vec{I}$ might be restricted. In order to decrease the size of series of measurements ($n$) to the required size $N$, discrete Haar Wavelet transforms [45] can be applied on the raw series of measurements. Thereby, $n$ must provide the following property: $n = N \cdot 2^k, k \in \mathbb{N}_0$.

The discrete Haar Wavelet transform is very simple and efficient and can easily be performed on a sensor node. The discrete Haar Wavelet transform is a digital filter consisting of a Low Pass Filter (LPF), which models the signal frequency, and a High Pass Filter (HPF), which models the noise. For data reduction only the LPF is considered.

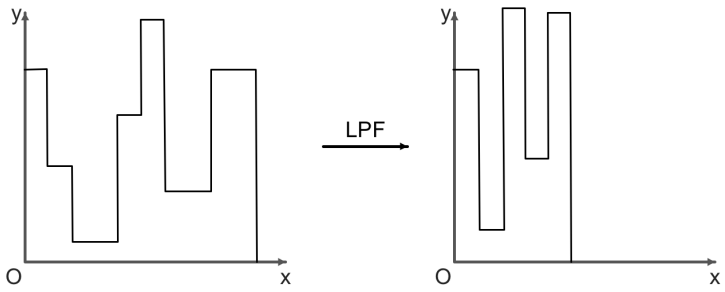



**Figure 3.15:** Haar Wavelet transform (LPF).

An example of a Haar Wavelet LPF is depicted in Figure 3.15. The LPF simply goes through a time series and computes the sum of any two subsequent values and divides the result by $\sqrt{2}$. Thus, a data reduction factor of two for each application of the LPF is achieved. In Figure 3.15, the 10 values of the input signal on the left are reduced to 5 values with the LPF. The LPF keeps low pass frequencies of a signal, while high pass frequencies are removed. Thus, a data compression is achieved by keeping the general shape of a signal.

In addition to the needed reduction in sample size, the Wavelet transform also smoothes the original input signal, which can either be interpreted as a de-noising of the original signal or as a generalization of the same.

### 3.6.7 Classification of Time-Discrete Events

So far, an introduction to classification problems and solutions by focusing on aspects and methods considered in our own approach has been given. The discussion of the relevant state of the art in classification with wireless sensor networks starts with approaches that classify events that are present as discrete entities in time. Most of these approaches apply statistical methods, but pattern recognition techniques as well as simple threshold-based mechanisms have also been proposed.

#### SensIt: Classification Functionality

Event classification with wireless sensor nodes has been addressed in the SensIt project, e.g., in [78], [131], [132], [31], [99]. A wide range of statistical methods has been covered in the SensIt project. Therefore, only work that has implications for this thesis is presented from this project here. In [78] the focus was on the localization and classification of sources based on their seismic fingerprint. As described in Section 3.4.2, the monitoring network is divided into space-time regions (grids) with at least one responsible leader node. Time-series of seismic measurements are gathered at the leader node(s). The authors propose three different classification algorithms to deal with the concurrent existence of multiple targets. These classifiers are: k-NN, maximum likelihood (MA), and support vector machines (SVM). The classifiers operate on the time-series associated with each event in a time-space cell. Limitations of the proposed statistical approaches are their rather centralized nature and their need for a considerable amount of data to provide statistically relevant results. The dependency on time-series implies delays.

Refinements of the proposed statistical methods have been proposed in [131], [132]. Statistical methods that compute the Maximum Likelihood (ML) of events with Expectation Maximization (EM) algorithms are proposed. In addition, numerical nonlinear optimization methods, which are based on Exhaustive Search (ES) and Multi-Resolution (ML) search, have been proposed. All approaches require a considerable amount of sensor readings to provide accurate results.

#### Classification Based on Local Binary Decisions

In [160] a fault-tolerant classifier based on local binary decisions has been introduced. Local binary decisions about an event are forwarded to a fusion center, which performs the final classification based on the collected data. A fault-tolerant fusion rule (classification) is applied. The classifier has been designed to consider faulty event reports. This means a certain number of wrong binary decisions are tolerated by the mechanism. To do so, wrong binary event reports are detected and corrected by error-correcting codes at the fusion center. The error-correcting codes are computed according to the frequencies of occurrence of errors. Accordingly, a priori knowledge of the events and pre-computation of the codes is required. Due to the pre-computation of the codes, the resulting classification rules are efficient. To

achieve good results, the fusion center requires a large number of binary decisions from network nodes, which imposes high communication costs.

The error-correcting code design is essential for this approach. In order to determine good codes, two algorithms have been proposed. The first algorithm is a cyclic column replacement approach, which is fast but can converge to a local optimum. The second approach is based on simulated annealing, which is a probabilistic metaheuristic to solve a global optimization problem. This mechanism is very robust as it can avoid local minimums, but imposes high computation costs.

### Dimensionality Reduction Based on Kernel Functions

In [43] the complexity of a classification problem is decreased by reducing the problem dimensionality. Thus, communication and computation costs can be lowered. The dimension reduction is performed similar to principal component analysis, which is a commonly used regression method. In detail, the problem reduction is based on kernel linear regression, where the kernel is represented as a weighted sum of local basis functions. However, the important property of the system is that a complex global function (e.g., a classification function) can be decomposed into tasks of lower dimensionality. The lower dimensionality tasks are then distributed among the network nodes. Thus, the nodes contribute to the global function by processing their local measurements. Accordingly, instead of communicating all measurements, only local constraints on the model parameters are negotiated. Thus, a dimensionality reduction is achieved, which decreases communication needs and fastens computations. In order to perform the decomposition, again a priori knowledge of the problem is required.

### Intrusion Detection and Fence Monitoring

Intrusion detection with wireless sensor networks has been investigated in [165], [32], [164]. In [165] initial results of fence monitoring with sensor nodes have been provided. Events such as a person climbing over a fence are collaboratively detected. Six different event sources have been distinguished based on their fence activation fingerprint. Event patterns are detected based on the activation of the accelerometer implemented on the sensor nodes. Both, node-level and collaborative classifications are based on thresholds and majority decisions. In subsequent work [32], [164] calibration of the sensor nodes and more advanced pattern recognition techniques has been introduced to improve accuracy. The node-level classification is divided into a learning phase, where event prototypes are learned, and into an operation phase, where subsequently occurring events are classified according to the learned prototypes. The node-level classification is based on the Euclidean distances between the observed event pattern and the stored event class prototypes. The collaborative event classification is based on consensus among the involved sensor nodes, i.e., a majority decision is performed. Only event types determined in the learning phase can be classified.

### Classification based on Fuzzy ART Neural Networks

Many pattern recognition mechanisms require the central storage of training data, which is then used for learning. Due to the availability of complete training data sets, these algorithms are able to provide high accuracy. On the other hand, they imply high communication and storage costs and require periodical re-computations. Adaptive Resonance Theory (ART) neural networks (see Section 3.6.5) avoid these drawbacks by learning new patterns online. Because no training data is locally stored, ART neural networks are in general not able to provide the accuracy of pattern recognition techniques that include training phases.

In [68], [67], [69] the application of ART neural networks for classification and dimensionality reduction in sensor networks has been proposed. Events are observed as feature patterns $\mathbf{x} = \{x_1, ..., x_N\}$. Each present pattern $\mathbf{x}$ is classified according to the local Fuzzy ART neural network. Only the resulting classification number is forwarded to a fusion center. Thus, local classifications based on Fuzzy ART reduce the reporting volume from $N$ to 1. The Fuzzy ART systems proposed in [68], [67], [69] are only used for data reduction by merging multiple sensor readings at discrete points in time. However, Fuzzy ART neural networks are well-tailored to anomaly detection problems. In our own work we extend Fuzzy ART neural networks to detect anomalies in time series of measurements. Moreover, we implement Fuzzy ART neural networks on node level, but use a binary ART neural network at the fusion center. Thus, anomalies can be efficiently detected. High compression rates on node-level are an appreciated side-effect of our systems.

## 3.6.8 Continuous Event Classification and Anomaly Detection

The classification algorithms proposed in the previous section have been introduced to address time-discrete classification problems. In this section the classification of events which evolve over time is addressed. These kinds of events are difficult to predict and by nature vary in duration of appearance. These classification problems mainly face two restrictions in sensor networks. First, processing power and memory are limited. Therefore, complex pattern classification methods are difficult to implement at the node level. Second, communication costs are high, which prevents the option of forwarding all raw sensor data to a fusion center.

The classification of continuous events has recently gained attention in wireless sensor network research. An adaptation of the approaches presented in the last section is difficult and in some cases not feasible. Apart from classification, the detection of abnormal events also belongs in this category. Unlike general classification tasks that are interested in specific events, anomaly detection systems are mainly interested in identifying behavior that deviates from expected behavior. Accordingly, anomaly detection simplifies the classification problem as long as only the distinction between normal and abnormal events is required. Relevant related work is provided in this section.

### Recognition of Bird Species with Neural Networks

The performance of bird species recognition by applying time delayed neural networks has been studied in [16]. Different preprocessing methods and different sets of features have been evaluated. A context-aware and neural network architecture has been designed which considers the dynamic nature of bird songs. The approach includes a noise reduction algorithm.

The approach requires relearning phases if new bird species appear. Moreover, each possible event type requires its own neural network, which imposes high storage demands on the sensor nodes. Optionally, the recognition of specific bird species could be restricted to responsible sensor nodes. Thus, the storage requirements could be lowered but more sensor nodes would be required.

### Tracking and Classification with Continuous Transferable Belief Models

The authors of [122] [113] propose the usage of a continuous Transferable Belief Model (cTBM) to classify continuously evolving events which are present to the system as a sequence of events. The cTBM is similar to probabilistic Hidden Markov Models, whereby the underlying physical process is not random. To track and classify events, cTBM has been combined with a particle filter. Each particle is used to construct a set of beliefs, which are then fused with the existing beliefs for classification. These new beliefs are fed back to the system to update the particle filter. The tracking group formation and the selection of nodes that report event data has been done according to [163]. Statistical methods are used to optimize the trade-off between communication cost and the quality of the information obtained by the sensors.

The approach is more robust than a similar Bayesian approach. However, implementing cTMB together with particle filters and dynamic programming for the tracking group formation imposes high communication and computation costs, making the approach not well-suited for tiny wireless sensor nodes.

### Intrusion Detection and Security

Intruder detection has gained much attention in the computer security community. Algorithms based on State Vector Machines (SVMs), Fuzzy Logic Controllers (FLC), Principal Component Analysis (PCA), Hidden Markov Models (HMMs), or Instance-based Learning (IBL), have been introduced. [107] gives an overview of existing techniques. Due to their complexity most of these algorithms cannot be simply adapted to sensor networks, though. Nevertheless, some first steps in the direction of applying intrusion detection systems (IDS) to static sensor networks have been done [123], [27], [65]. In [123] general guidelines and requirements considering an integration of IDS in wireless sensor networks are provided. [27] and [65] apply rule-based voting schemes to prevent certain kinds of network attacks.

Instance Based Learning (IBL) based on observed and stored profiles [46] is a promising approach for intruder and anomaly detection. However, the required

storage of profiles soon exceeds the memory space of tiny sensor nodes. Therefore, a manageable representation of the profiles is needed, which could be achieved with ART neural networks. Fuzzy Logic Controllers (FLC) have also been considered for the detection of abnormal behavior, e.g., in [22]. These approaches are very lightweight, but require accurate event modeling by experts, which makes the approaches inflexible.

### Intruder Detection in Building Surveillance

Recently the authors of [86] proposed an intruder detection system that combines Fuzzy ART mechanisms according to [69] with Markov chains. The approach combines wireless sensor networks with mobile robots. The mobile robot is the leader of the system. Upon detection of critical behavior, the robot is informed by sensor nodes to travel to the location of interest. The robot is equipped with additional hardware such as a camera and can provide more detailed information than the sensor network. In a learning phase the sensor nodes learn normal behavior, i.e., they train their Fuzzy ART neural network to identify known behavior. Unlike the approach proposed in [69] event patterns that evolve over time are considered. This is achieved by integrating the Fuzzy ART neural network with Markov chains that model state changes over time.

The proposed approach learns normal behavior and determines anything that deviates from that known behavior as an anomaly. After the learning phase, the operation of the system is static. If new event patterns have to be learned, the learning phase must be repeated. This makes the algorithm rather inflexible, because it loses its online learning capability. In our own work, we allow our Fuzzy ART neural network to refuse rarely used event patterns if the learning buffer is full. Thus, new event patterns can always be learned at the cost of losing rarely used knowledge. Because the detection of anomalies is the main scope of our system, temporarily discarding sporadically matched event patterns can be tolerated. The goal of [86] differs largely from our goal. In [86] the Fuzzy ART neural networks implemented on the sensor nodes learn normal behavior. In operational mode the sensor nodes aim to recognize learned normal behavior. In our own approach nothing is learned in advance. Sensor nodes have a small short term memory to temporarily remember common behavior. Anything uncommon is then reported to a base station.

### Anomaly Detection in Underground Coal Mines

In the work of [161] spatiotemporal anomalies in gas distributions in underground coal mines are detected and reported by Bayesian networks. Bayesian networks are a generalization of Hidden Markov Models (HMMs), where not only the present and the last system state, but $k$ past system states, where $k > 1$, are considered for classifications.

The sensor nodes used in [161] are wired and therefore face much lower communication constraints. The detection of critical events is very important because

non-detection might endanger the miners. Therefore, the usage of communication and computation intensive Bayesian networks is justified. However, such algorithms pose too high of a demand on general wireless sensor networks and are therefore only applicable to customized applications.

#### Anomaly Detection Based on Artificial Immune Systems

Anomaly detection has furthermore been addressed by Artificial Immune Systems (AIS) [28], [96], [97]. In [28] an early approach that detects anomalies in time series by using ideas from immunology has been proposed. In [97], [28] these ideas have been taken up to improve security in mobile ad-hoc networks. AIS systems use the paradigm of self/non-self discrimination and provide incremental online learning ability. Thus, they resemble Fuzzy ART systems. Unlike Fuzzy ART neural networks, which implement a positive selection algorithm, AIS systems implement a negative selection algorithm. Moreover, AIS systems are probabilistic methods, whereas Fuzzy ART systems are special kinds of instance-based learning algorithms. AIS systems are in general less compact than Fuzzy ART systems and accordingly require more memory.

### 3.6.9 Threshold-based Event Classification

Finally, we introduce some state-of-the art of threshold-based classifiers. Due to their threshold-based nature, such systems can be applied to both discrete and continuous event classification. All approaches require expert knowledge. Therefore, no unsupervised learning is possible. Sensor nodes can easily be updated with new functionality by writing and downloading new configurations.

#### Querying Systems

Some querying systems have been adapted to support event detection (e.g., TinyDB [93], [91], [92] and Cougar [171]). Event queries are downloaded and run on the sensor nodes. A recent query-processing mechanism that addresses event detection in specific areas has been proposed in [124]. Event properties such as event diameter, expected occurrence time and expected sensor activations as well as the according parameters are defined by a customized declarative query language. For these definitions and declarations a system expert is required. The approach presumes a priori knowledge of the occurring events and thresholds in order to determine event boundaries. This prevents dynamic anomaly detection, though.

In [170] the problem of exploring relationships between sensor data readings in specific time windows is addressed. Such specific time windows are typical for querying systems. It is shown that the detection of events in a time window using the common aggregate or selection queries is difficult. The same applies to classification problems. The authors address the problem of processing window self-joins in order to detect events of interest. The approach might be helpful to increase

accuracy and organize the event observing area more effectively. Nevertheless, the definition of events and the according queries still requires expert knowledge.

#### Lightweight Detection and Classification in Military Environments

A classification algorithm for EnviroSuite [1] has been proposed in [42]. Due to the project goal of tracking military units, the classification and differentiation of three different event types has been investigated, namely of vehicles, persons, and persons carrying ferrous objects. The classification of these three classes is performed efficiently by using a hierarchical classification architecture that is based on settable thresholds. Thus, individual classification steps are performed at different levels of the system. Signal processing and local decisions are done at node level. Cooperative event decisions are then determined in the tracking groups.

The proposed approach is not declared as a typical querying system. Nevertheless, it is very similar, because thresholds can be set according to configuration contexts that are distributed in the network. Like all query and scripting based mechanism the expected events have to be modeled by a system expert. The detection and classification is restricted to three event types.

#### Classification Based on Decision Trees

In [8] classification problems are addressed by a multi-tiered classifier based on decision trees. Decision making is performed by successively querying nodes in the tree. Sensor nodes are dynamically activated and the sampling rate is dynamically adjusted such that at any given time point only the data needed for classification is collected. Thus, the trade-off between accuracy and power usage can be optimized. Patterns of moving persons are classified by a wearable gait monitoring system.

The system has shown to provide similar accuracy as a support vector machine (SVM) approach, while requiring less power. The construction of the decision tree and the querying of the tree again requires expert knowledge, which makes the approach static and inflexible.

## 3.7 Monitoring Applications

In this section we give a brief overview of some related work in event monitoring systems for wireless sensor networks. The section is divided into common environmental monitoring applications and specific building and structural health monitoring applications, which deploy video surveillance technology.

### 3.7.1 Environmental Monitoring

Many sensor networks have been designed for outdoor monitoring purposes, especially environmental monitoring and animal monitoring. The range of applications includes but is not limited to seabird habitat monitoring [94], cattle control [133],

zebra herd monitoring [174], the monitoring of coal mines [79], [161], water monitoring and flood detection [6], volcano monitoring [162] and glacier monitoring [5]. Some of these approaches provide data aggregation within the network while others route all raw data to a base station for later analysis. Some of these works have influenced the development of our work. However, our event detection system differs from these approaches in many ways. In particular system-specific requirements need to be considered.

### 3.7.2 Visual Sensor Networks

Building monitoring and structural health control have been addressed by visual sensor networks. These networks integrate wireless sensor network technology with wireless video surveillance. Due to their potential, visual sensor networks have gained much attention in recent years. Much effort has been put into tailoring video streaming techniques to the requirements of resource-constrained systems. Efficient video coding [47] [74] and reliable routing [21] have been addressed as well as video calibration and deployment [70]. A dual-camera sensor network has been proposed in [168]. An energy-efficient persistently running low resolution camera triggers a high resolution camera on demand. Security and privacy issues concerning the usage of visual sensor networks have been considered in [89].

In [7] an event-based triggering system has been used for structural health monitoring of bridges. A wireless sensor network is deployed and used for video camera control. If the wireless sensor network detects an abnormal event such as ”large structure tilt detected”, a video camera is activated and zooms into the area of interest. The application goal of the system resembles our own one. However, events are thrown based on thresholds. These thresholds have to be determined by system experts and work only for the specific deployment. In contrast, our system learns and determines abnormal behavior unsupervised. Moreover, in-network processing is supported due to the usage of adaptive memory. Our system is currently run standalone. However, it would be possible to use our energy-efficient anomaly detection system to trigger a more energy consuming system such as a wireless video surveillance system on demand, i.e., if some suspicious (abnormal) office occupancy is reported.

## 3.8 Sensor Node Platforms

This section introduces the two wireless sensor node platforms that have been used during the development of our event detection system.

### 3.8.1 The Embedded Sensor Board

The ESB sensor boards [127] have been used for the experimental evaluation of our event detection system. ESB nodes consist of a TI MSP430 microcontroller, 2kB of RAM, a 60kB flash memory, and a low power consuming radio transceiver

(868MHz) operating at a bandwidth of 19.2kb/s by default. For some functionality such as the communication scheme of DELTA that requires short periods with high message load, a bandwidth of 19.2kbps is too small, because it introduces collision probabilities that are too high. The ESB sensor node software has therefore been changed to run with ASK modulation and 76kbps. This setting provides a good trade-off between energy efficiency, bandwidth and reliability. Furthermore, the sensor nodes are equipped with a number of sensors such as luminosity, temperature and vibration. The ESB boards face two main restrictions. First, the bandwidth is comparatively low. Second, resources in terms of memory and processing power are limited. Both limitations are basically caused by the miniaturization of the implemented hardware. The ESB nodes have to work at a maximum of 3V DC. Furthermore, as little energy as possible should be consumed to extend node lifetime.

**Figure 3.16:** ESB sensor node.

An ESB sensor node is depicted in Figure 3.16. ESB nodes operate in the 868 MHz frequency band. The transmission range is approximately 37 m, whereas the interference range is approximately 52 m. The data rate is set to 115.2 kbps. The transmission power of ESB sensor nodes is 0.75 mW and the receiver sensitivity is -95dBm. The energy consumption in transmission mode is 5.2 mA. Idle listening and receiving both require about 4.7 mA, while the radio only needs 5 $\mu$A in sleep mode. The parameters of the sensor nodes in our simulations have been configured according to values from the ESB nodes.

In the real-world experiments of DELTA the TSL245 light sensor [55] implemented on the ESB sensor nodes has been used. The output frequency of the TSL245 sensor is shown in Figure 3.17. The provided light measurement software supports only binary decisions (light on/off) for efficiency reasons. For detection and tracking in our context this is not appropriate, though. Therefore, we have reimplemented the software. The light sensor is associated with an interrupt-capable register. On each positive edge of the output frequency of the TSL245 an interrupt is thrown. In the interrupt routine a counter is incremented. The costs for this solution increase with the irradiance. Therefore, the maximum spectrum is limited

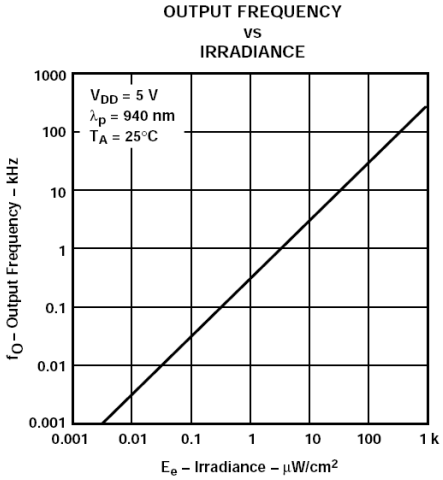



**Figure 3.17:** Output of the TAOS TSL245 infrared to frequency converter [55].

to a frequency of 100 kHz. Each higher output frequency is considered to be the maximal brightness. The output frequency of the TSL245 on a desk in a normal office during the day is approximately 2kHz.

### 3.8.2 The TmoteSky Platform

TmoteSky sensors consist of a microprocessor, some memory, an IEEE 802.15.4 compliant radio, and a number of sensors such as temperature, humidity, and light. The radio operates in the 2.4GHz frequency and can therefore interfere with standard IEEE 802.11b wireless networks. Data rates of 250 kbit / s can be achieved with the radio. Light can be measured with two sensors. The first light sensor measures only the photosynthetic active radiation (PAR), i.e., the visible light with a wavelength between 320 and 730 nm. The second light sensor measures the total solar radiation (TSR), including infrared, ranging from 320 to 1100 nm. The TmoteSky sensor node is depicted in Figure 3.18.

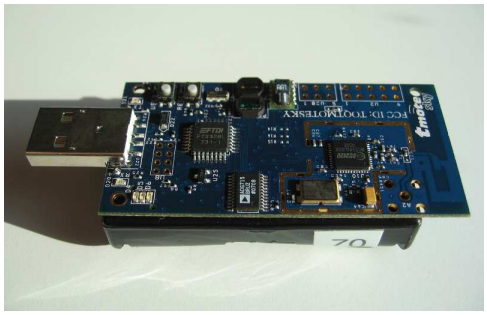


**Figure 3.18:** TmoteSky sensor node.

## 3.9 Conclusions

In this chapter relevant work from the research areas of medium access, routing and topology control, event detection and tracking, and event classification and anomaly detection have been presented. In the last section a number of monitoring systems have been introduced. Since none of these monitoring systems is tailored to office monitoring, none of these systems is optimally suited to address our application requirements, which are the support of long-term deployments and the satisfaction of event detection requirements (e.g., the required accuracy).

Even though many problems have been solved with existing approaches, no event detection architecture that is laid out for long-term operation on tiny sensor nodes has been presented so far. Numerous classification and anomaly detection procedures have been proposed. However, all these mechanisms either solve problems that are not addressed in our context or they apply methods that are too resource consuming to support long-term deployments. However, a long-term deployment of our event detection system is crucial because continuous building monitoring is required. Periodic physical battery replacements are not an option. Our event and anomaly detection features implement some functionality used in related work too. For example, ART neural networks have been previously used in wireless sensor networks too. However, the common ART neural network system design needs to be extended to meet the specific requirements of our system.

Our system provides a nonlinear localization and signal strength estimation method that optimizes the trade-off between communication minimization and localization accuracy. Current state of the art focuses on very accurate estimations. Comparatively high communication load is accepted. However, approximate estimations are sufficient to support classification. Our method outperforms the state of the art in minimizing communication load.

The current state of the art in event detection and tracking does not adequately address communication minimization and detection accuracy. There are approaches that either optimize group organization or they optimize detection accuracy. Our DELTA detection and tracking system addresses both requirements. Thus, DELTA outperforms previous work in finding an optimal trade-off between network organization costs and detection accuracy.

Finally, we provide our event detection systems with medium access and routing. In particular, we have proposed a mechanism that implements routing on the MAC layer without requiring additional control traffic. Thereby, synchronized contention-based MAC protocols have been integrated with connected dominating set theory. Such integration has not been found in related work yet. In addition to the routing support, extra energy can be saved because non-backbone nodes can be temporarily turned off.

To meet the requirements of our target application, a tailored event detection system has been developed that integrates different network layers. Our approach gains from results obtained in related work, but also extends particular related work. In addition, novel features and an integrated system design are provided.

Chapter 4

# Local Clock Synchronization

In this chapter a mechanism to achieve common listen/sleep cycles in energy-efficient synchronized contention-based MAC protocols is proposed [180], [158]. Synchronized contention-based MAC protocols follow periodic listen/sleep cycles. The protocols face the problem of virtual clustering if different unsynchronized listen/sleep schedules occur in the network, which has been shown to happen in wireless sensor networks (see Section 3.2.1). To interconnect these virtual clusters, border nodes, which maintain all locally relevant listen/sleep schedules, are required. This is a waste of energy if a common schedule can be locally determined. To achieve this common schedule, we propose a local synchronization mechanism that makes use of gravitation. Clusters represent the material, whereas synchronization messages sent by each cluster represent the gravitation force of the according cluster. Due to mutual attraction all clusters merge. The synchronization mechanism itself is not altered. Every sensor node competes for transmission of synchronization messages as it normally does. Accordingly, no overhead is introduced by our algorithm, but a not yet used property of synchronization mechanisms is exploited. This local synchronization mechanism is used in the MAC protocols, which are implemented to provide our topology control and routing backbone algorithms with the required medium access functionality.

## 4.1 Introduction

Synchronized contention-based MAC protocols maintain low duty cycles. This means the sensor nodes follow periodic listen/sleep cycles. In the listen cycle the sensor nodes are able to communicate with neighbor nodes and can forward pending data. In the sleep cycle they shut down their radio to preserve energy. In order to synchronize their listen/sleep cycles with neighboring nodes, SYNC messages are periodically exchanged. Through this synchronization process, nodes which maintain the same listen/sleep cycle are organized into clusters. This synchronization of common listen/sleep cycles is called virtual clustering (see Section 3.2.1). To support communication between different clusters, border nodes which interconnect the according clusters are required.

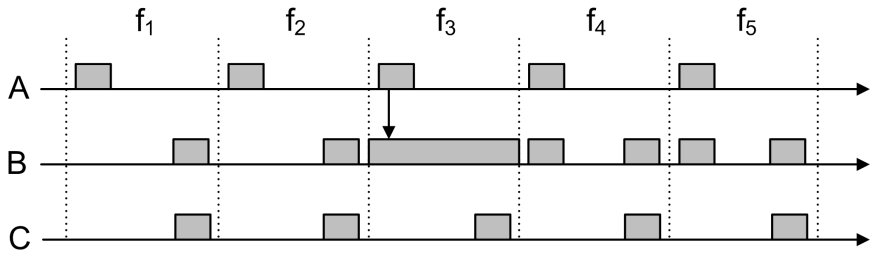



**Figure 4.1:** Periodic sleeping and virtual clustering.

The synchronization mechanism implemented in synchronized contention-based MAC protocols is illustrated in Figure 4.1. Node A is member of a virtual cluster, whereas nodes B and C are members of another disjoint virtual cluster. This is indicated by the disjoint listen periods (colored gray) in Figure 4.1. All nodes could be in transmission range of each other. However, in the example above, it is only required that node A can hear node B and node B can hear both nodes A and C. From time to time each node remains awake for an entire frame length $f_i$ in order to scan for present schedules. In Figure 4.1 this is node B in frame $f_3$. In this frame B learns the cluster of node A, because it overhears the SYNC message sent by node A. Only this SYNC transmission is shown in Figure 4.1. Node B becomes a border node as it interconnects two clusters. This means node B synchronizes to both known schedules henceforward, i.e., beginning in frame $f_3$.

Experiments have shown that four different virtual clusters have already evolved in a multi-hop network consisting of 50 nodes running S-MAC (Section 3.2.1). Moreover, it has been shown that border nodes had to listen to up to three different schedules. In all four experiments more than 44% of all network nodes followed at least two schedules. In two of the four experiments 34%, respectively 47%, of all network nodes even had to listen to three virtual clusters. Thus, the border nodes have higher average energy consumption than normal cluster nodes. The problem of virtual clustering has further been illustrated in Fig 3.2.

While the authors of [85] have used a global mechanism to solve the problem (see again Section 3.2.1), we propose a local adaptive clock synchronization scheme that achieves local synchronization. A global solution requires system-wide synchronization towards one global schedule. This implies overhead in terms of signaling and requires the storage of fallback mechanisms, i.e., of temporary valid local schedules. Another possibility to discharge border nodes would be to alter the respective role amongst neighbor nodes. However, such a role circulation would require additional communication and organization. LACAS avoids these drawbacks. Maintaining a global schedule is unnecessary, because of the locality of communication links between network nodes. LACAS avoids the drawback of virtual clustering and leads to a uniform distribution of the energy required for synchronization.

## 4.2 Local Adaptive Clock Assimilation Scheme (LACAS)

LACAS implements a mechanism similar to gravitation. In terms of a virtual clustering problem, this means that larger clusters attract smaller ones more than vice versa, until the clusters finally merge. In LACAS, the cluster nodes represent the mass, and the number of sent SYNC messages represents the gravitation force. Because all sensor nodes implement the same contention-based transmission scheme, large clusters broadcast on average more SYNC messages than small ones and thus cause more attraction.

LACAS only exploits the information exchanged by synchronization messages. Therefore, no additional control traffic is generated. Moreover, the loss of SYNC messages does not affect the principle of LACAS, but only temporarily decreases the gravitation force of a cluster.

**Synchronization of LACAS**

```
while true
    Compete for SYNC transmission in every listen period;
    if scanning == true          // periodically true in every 32th listen/sleep cycle
      Remain awake for the whole cycle;
      Scan for unknown listen schedules;
      Span own listen schedule over all learned (known) schedules;
    end
    if SYNC overheard == true
      Adapt the own listen schedule to the schedule of the SYNC sender for α;
    end
```

The LACAS synchronization procedure is described in the pseudo code above. Every network node performs the described actions. This means it periodically scans for unknown listen/sleep schedules. If any unknown listen/sleep schedule is overheard, the respective sensor node becomes a border node by spaning its own listen schedule over all known listen/sleep schedules. Thus, the node is ensured to overhear the SYNC messages from all involved clusters. Every overheard SYNC messages causes the adaptation of the own listen schedule for some value $\alpha$ (e.g., $\alpha$ = 5%). The parameter $\alpha$ controls the attraction caused by a SYNC message. Only the border nodes attract clusters directly. According to the pseudo code above, in the first step of a merging process, the schedule of a border node is expanded and then it starts to contract again. Over time, all sensor nodes converge towards a common listen/sleep schedule. This is due to gravitation force of clusters.

A merging process is shown in Figure 4.2. The gray bars indicate listen periods. The white bars describe adaptations. Node C stays awake for a whole listen/sleep schedule in $f_2$. Having detected another schedule, it spans its listen period over both known schedules (see $f_3$). This listen period contracts then to the normal schedule length merging both connected clusters. The parameter $\alpha$ controls the gravitation force. High values for $\alpha$ lead to high attractions and fast convergence.

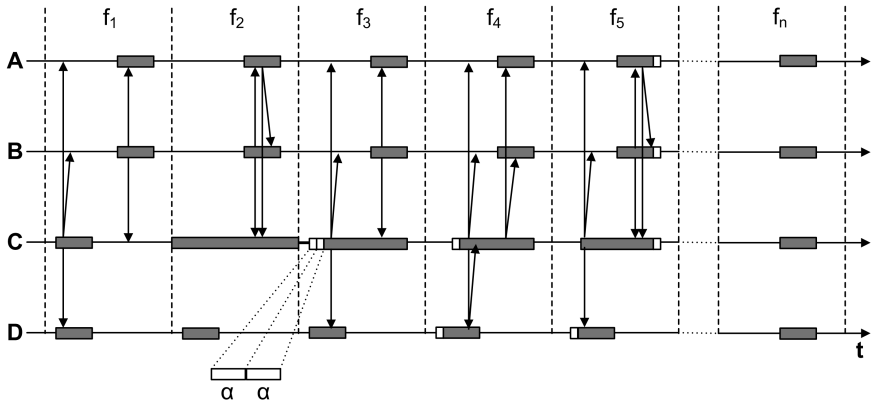



**Figure 4.2:** Gravitation principle of LACAS: Cluster detection and merging.

Thus, connections between clusters can temporarily be broken. This affects only the convergence time of the clusters, though. The gravitation mechanism itself is not compromised. In the worst case, nodes are successively transferred from the smaller cluster to the larger one. The problem is discussed later. The contraction of the schedule of node C continues after period $f_5$ and ends in period $f_n$.

Initially nodes A and B form cluster 1, while nodes C and D form another cluster 2 (see frame $f_1$ in Figure 4.2). Because T-MAC is used, all nodes periodically stay awake for an entire frame $f_i$ in order to detect other clusters. In Figure 4.2 this is node C in frame $f_2$. Having learned both clusters, C spans its own listen period over both schedules. Moreover, as it has received two SYNC messages from nodes A and B from cluster 1, node C moves its listen period for $2\alpha$ towards the listen period of cluster 1 (frames $f_2$ and $f_3$ in Figure 4.2). In frame $f_3$, C is able to transmit its own SYNC message and receives another one from node B from cluster 1. Accordingly, the listen period of node C again moves for $\alpha$ towards the listen period of cluster 1. Having received a SYNC message from node C, node D is attracted too ($f_4$ in Figure 4.2). In frame $f_4$ node C is able to transmit a SYNC message in both schedules. Accordingly, cluster 1 and node D are attracted towards C. C itself moves towards D as it has received a SYNC message from D (frame $f_5$ in Figure 4.2). The merging process continues in Figure 4.2 after frame $f_5$. After a while, both clusters will fuse. In the example, cluster 1 transmits in general more SYNC messages than cluster 2 (it has three members, whereas cluster 2 has only two). Thus, both clusters will merge closer to the original schedule of cluster 1.

The adaptation parameter $\alpha$ is crucial for performance. A small $\alpha$ implies a long merging period. On the other hand, a large $\alpha$ leads to a fast convergence towards a large cluster, which might disrupt the connection between a border node and its smaller cluster. This disconnection is not a major problem, because the clusters are connected again when a border node remains awake for a whole frame, but it increases merging time. In the worst case, nodes pass successively from the smaller cluster to the larger. The convergence of LACAS is however not affected.

In general growing clusters have a growing number of border nodes. Thus, their gravitation force increases too. In the current deployment we have chosen a value for $\alpha$ of 5%. Thus, all sensor nodes are able to synchronize within a few minutes. Expecting a network lifetime of at least several months, the synchronization time seems tolerable. Finally, only the listen periods of the according MAC protocol are optimized. Any subsequent data exchange period is not affected by LACAS.

## 4.3 Evaluation

LACAS has been evaluated on top of T-MAC in simulations. All network nodes wake up randomly within the first 30 simulation seconds, and immediately begin to synchronize. T-MAC enhanced with LACAS has been used as MAC protocol for the topology control algorithms presented in Section 5.2. Every simulated network has been synchronized within a few minutes. Moreover, the synchronization remained stable in all simulations. Because the development of LACAS is part of a cross-layer approach for topology control [157], the performance of LACAS has been evaluated using a larger scenario. The topology control mechanism establishes a routing backbone after a synchronization and neighborhood learning period of 300 s. Two approaches to implement the backbone are discussed in the next section. Non-backbone nodes disconnect from the network and go to sleep for a longer period. This has some impact on the convergence time of LACAS as fewer SYNC messages are sent due to the temporal unavailability of the non-backbone nodes. However, the principle functionality of LACAS is not affected.

### 4.3.1 Simulation Scenario and Parameters

The parameters of T-MAC have been set according to [142]. All relevant simulation parameters are listed in Table 4.1. All nodes follow a periodic listen/sleep frame of 610 ms, of which they are awake for at least 13.5 ms, i.e., if no data transmission is pending. This minimum wake-up period consists of the synchronization period, which is 7 ms, and the traffic-adaptivity period TA, which is required by T-MAC and has a duration of 6.5 ms. Each node remains awake for a whole frame in every 35th frame, i.e., every 21.35 s. This is required in order to detect neighbor nodes which follow different listen/sleep cycles.

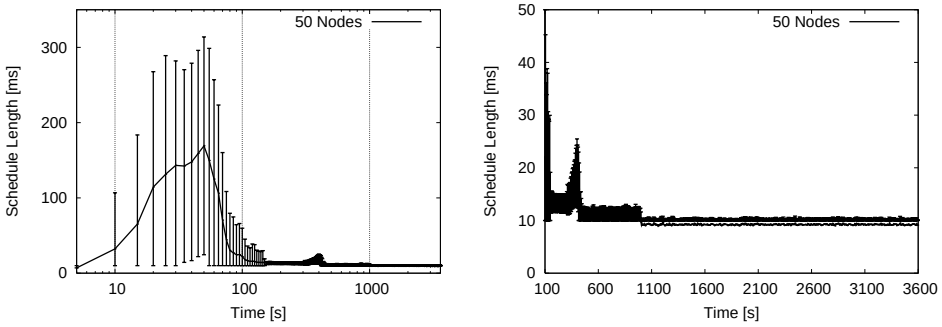
Three different network sizes of 50, 100 and 200 nodes have been simulated. The respective simulation areas are 18'000, 36'000 and 72'000 $m^2$. Considering the different simulation areas, their respective population, and the transmission range of approximately 37 m (see Section 3.8.1), an average node density of 12 neighbors is obtained. The network topology was randomly generated taking network connectivity into account. Any experiment has been repeated 20 times. The spectrum of the schedule lengths present at a specific time point is indicated by the standard deviation. The properties of the sensor nodes are configured according to values from the Embedded Sensor Board (ESB) platform (see Section 3.8.1).

**Table 4.1:** Parameters of the MAC simulations.

| Parameter | Value |
|---|---|
| Listen/Sleep Frame | 610 ms |
| SYNC period | 7 ms |
| TA | 6.5 ms |
| Periodic wake frame | 21.35 s |
| Network size | {50, 100, 200} |
| Simulation Area | {18'000, 36'000, 72'000} $m^2$ |
| Network density | 12 neighbors |
| Node deployment | random, but guaranteeing connectivity |

### 4.3.2 Convergence of Schedule Length with LACAS

In this section the convergence time of LACAS is investigated. The independent initial wake up of the network nodes in the first 30 s of the simulation leads to multiple coexisting schedules in the beginning. The evolution of the schedule length of each network node has been monitored over the first hour of operation and the schedule length has been captured every 5 s. The evolution of the mean common schedule length in a network consisting of 50 nodes is shown in Figure 4.3.



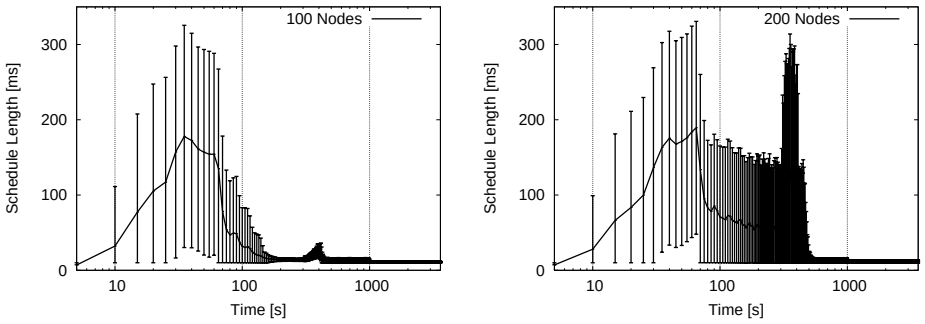

**(a)** Evolution in the first hour (log-scaled). **(b)** Evolution ignoring the first 100 s.

**Figure 4.3:** Schedule length convergence in a network consisting of 50 nodes.

Figure 4.3(a) shows the evolution of the schedule length over the whole first hour of operation. The schedule length converges to a length of approximately 13 ms within the first 200 s. Of course, in this convergence period the distribution of the schedule length is high in the network. There are nodes that follow common schedules and thus already have a short schedule length. On the other hand, there are numerous nodes interconnecting different schedules, which results in a temporarily increased average schedule length.

Figure 4.3(b) shows the evolution of the mean schedule length after the first

100 s. The peak at 400 s is due to the backbone scenario as described above. This has two reasons. At 300 s the parameter $\alpha$ is adapted from 0.05 to 0.5 to achieve faster convergence. This leads to the temporary peak. After the adaptation the performance improves slightly. Sleeping non-backbone nodes lead to a smaller amount of SYNC messages, which further reinforces the effect. The peak is on the order of a duplication of the schedule length. The adaptation of $\alpha$ could be implemented in LACAS without cross-layer optimization too. The mean schedule length converges to 13 ms without adaptation and to 10 ms with adaptation. The average schedule length remains stable after 200 s without adaptation and after 450 s with adaptation.




**(a)** Network consisting of 100 nodes.

**(b)** Network consisting of 200 nodes.

**Figure 4.4:** Schedule length evolution in larger networks (log-scaled).

Figure 4.4 shows the impact of the network size. The performance of LACAS in a network consisting of 100 nodes is depicted in Figure 4.4(a). The performance is very similar to the performance in the network consisting of 50 nodes. However, convergence takes longer for the network consisting of 200 nodes (Figure 4.4(b)). Compared to an intended network lifetime of several months or more, this delay is still insignificant. The increased convergence time of LACAS with network size is due to the hop-by-hop impact of the gravitation principle. Thus, clusters show an impact similar to the movement of a ripple through water over multiple hops.

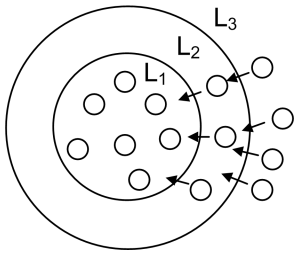



**Figure 4.5:** Ripple effect of gravitation over multiple hops.

The effect is illustrated in Figure 4.5. Clusters of nodes such as the nodes in $L_1$ attract nodes at the boundaries. The nodes in $L_2$ again have an impact on their border nodes, i.e., on the nodes in $L_3$. The effect causes complex mutual influences. Moreover, due to the increased time needed for dissemination, clusters located far away from each other have a longer lasting impact on each other than nearby clusters. The probability of presence of such clusters grows with network size. Thus, the convergence time increases with network size too.

LACAS is not able to converge to a short schedule length before cross-layer adaptation occurs for a network size of 200 nodes (see Figure 4.4(b)). The cross-layer impact again leads to a temporary duplication of the average schedule length, which in this case is much longer. However, the schedule length converges quickly to 10 ms after the adaptation. This fast convergence is due to the cross-layer approach. Without optimization the convergence would look similar to Figures 4.3(a) or 4.4(a) without a peak. It would only require some more time.

The average schedule length has converged in all evaluated network topologies and sizes to a length of approximately 10 ms. This is not surprising, because the convergence (gravitation) is basically a local process. Mainly local communications have an impact on local convergence and any local communication is independent of the network size. On the other hand, due to the ripple effect it is not possible to achieve the schedule length of 7 ms of T-MAC. There is thus a trade-off between avoiding the border nodes and the achievable schedule length. On the node level, every border node with disjoint schedules consumes more energy than any node running LACAS. In terms of the overall energy consumption, LACAS preserves energy if the following inequation applies ($\forall i : n_i \in \mathbb{N}$):

$$n_1 \cdot 7ms + n_2 \cdot 14ms, + ... + n_k \cdot k \cdot 7ms > n_t \cdot 10ms; \quad \sum_{i=1}^{k} n_i = n_t \qquad (4.1)$$

where $n_i$ is the number of nodes maintaining a given number of schedules and $n_t$ is the total number of nodes in the network. Inequation (4.1) assumes that the different schedules are disjoint. Otherwise, overlaying schedules would need to be included. Unlike virtual clustering, LACAS achieves a uniform charging of the batteries, thereby avoiding quick depletion of specific border nodes. Therefore, LACAS might even be favored over virtual clustering if the average energy consumption is worse. The expected energy consumption of virtual clustering and LACAS in real-world networks is discussed in the next section.

### 4.3.3 Analysis of Power Consumption

Unlike the convergence of LACAS, realistic virtual clustering is difficult to simulate. Virtual clustering mainly occurs due to physical impacts such as communication gray zones [175] and temporary unavailable communication links. These impacts depend on the used hardware and on the environment. Thus, they are difficult to simulate properly. Approximating those impacts in simulations might falsify

the simulations rather than improve them. Finally, these effects have little impact on the convergence of LACAS due to the robustness of the gravitation principle.

In order to assess the power consumption saved by LACAS we adopt real-world results obtained from experiments performed in [85]. In these experiments, multiple coexisting virtual clusters have been detected. The percentage of network nodes maintaining a certain number of schedules are listed in Table 4.2. The costs of T-MAC and LACAS are computed according to these values and inequation (4.1). The long-sleep impact of non-backbone nodes has not been considered in this evaluation, because it is based on a cross-layer optimization. Accordingly, all nodes follow a periodic listen/sleep schedule. The sensor network in [85] consisted of 50 nodes running S-MAC (see also Section 3.2.1).

**Table 4.2:** Percentage of coexisting schedules (taken from [85]).

| | Number of Schedules | | | |
|---|---|---|---|---|
| | **1** | **2** | **3** | **4** |
| Exp. 1 | 56% | 44% | - | - |
| Exp. 2 | 32% | 68% | - | - |
| Exp. 3 | - | 66% | 34% | - |
| Exp. 4 | 9% | 44% | 47% | - |

The results would be the same if T-MAC had been used due to the identical synchronization mechanism. As mentioned above, ESB nodes need 4.7 mA in idle listening state. We use this value to estimate the power consumption of LACAS. Furthermore, we assume that the different schedules, which evolved in the experiments in [85], are disjoint (see inequation (4.1)). SYNC messages that would have to be sent in the synchronization periods are not considered. However, a synchronized contention-based MAC protocol with virtual clustering would transmit more SYNC messages than a similar approach implementing LACAS due to the existence of border nodes. Table 4.3 shows the power consumption of T-MAC and LACAS to maintain all schedules of all network nodes in one listen/sleep cycle. The results apply as soon as the networks are stable, i.e., after the convergence to the common schedule length of 10 ms in the case of LACAS, after all virtual clusters have evolved in the case of T-MAC. Therefore, the values in Table 4.2 can be used. The power consumptions of T-MAC and LACAS are computed according to inequation (4.1) and the values in Table 4.2.

LACAS maintains only one schedule. Therefore, the expected power consumption of LACAS is the same in all four experiments. The estimations in Table 4.3 show that in a network consisting of 50 sensor nodes, depending on the experiment, more or less power can be saved with LACAS, i.e., between 0.02 and 1.57 mAs in one listen/sleep cycle of 610 ms. Even though LACAS has a slightly longer minimal schedule length than T-MAC, LACAS is estimated to perform at least as well as T-MAC in all four experiments. Considering a network lifetime of months

**Table 4.3:** Power consumption (in mAs) per listen/sleep cycle.

| | T-MAC | LACAS |
|---|---|---|
| Exp. 1 | 2.37 | 2.35 |
| Exp. 2 | 2.76 | |
| Exp. 3 | 3.85 | |
| Exp. 4 | 3.92 | |

or more, the possible energy savings are promising. The estimations shown in Table 4.3 concern the average energy consumption over all nodes (in one listen/sleep cycle). However, border nodes with disjoint schedules consume more energy than the average and would therefore run out of energy sooner than other network nodes. On the other hand, LACAS distributes the energy consumption load more effectively. Therefore, even in the experiments where virtual clustering consumes on average only slightly more energy than similar runs with LACAS, network lifetime might be significantly extended with LACAS. Finally, LACAS is very stable and robust and requires the storage of only one schedule.

## 4.4 Conclusions

In this chapter a simple local clock synchronization scheme has been proposed. LACAS provides system-wide local clock consistency and therefore avoids the drawback of virtual clustering. It has been shown that the overhead of LACAS is marginal. Moreover, the synchronization procedure converges fast, i.e., within minutes for the simulated networks and remains stable thereafter.

The fast convergence of the algorithm has been shown in simulations. LACAS exploits the information exchanged by SYNC messages. If messages are lost, the functionality of LACAS is not affected. The concerned cluster only shows currently lower attraction. In related work it has been shown that variations in radio range and temporary unavailability of the radio have a high impact on the presence of virtual clusters. On the other hand, real-world properties have less impact on LACAS, because the gravitation mechanism is very robust. Therefore, the energy consumption of LACAS has been estimated based on real-world results, which have been collected in related work. LACAS has shown to preserve energy in comparison to virtual clustering. Moreover, the energy load distribution is better with LACAS. Thus, no nodes are more charged than others which prevents possible network partitions due to nodes being depleted earlier.

To save energy and to prevent the depletion of heavily charged border nodes our event detection architecture is provided with synchronized contention-based MAC protocols that implement LACAS.

Chapter 5

# Backbone Support

This chapter introduces approaches and features to provide our event monitoring framework with energy-efficient medium access and routing [151], [148], [180] [157]. All mechanisms are implemented on the MAC and routing layers as illustrated in the system architecture overview in Figure 1.1. Nodes that are currently not required for routing are identified and temporarily disconnected from the network to save energy. The focus of all developments is on energy savings in the context of medium access and networking issues.

## 5.1 Introduction

The event detection system proposed in this thesis has the main goal of accurate long-term event monitoring and reporting. This in particular involves appropriate event detection, tracking and classification mechanisms on the application layer. Nevertheless, in order to optimize energy savings, the system needs some efficient medium access and routing functionality. In the last chapter we have provided a mechanism to optimize synchronized contention-based MAC protocols. In this chapter we will further exploit such MAC protocols and enhance them to support routing directly on the MAC layer. The synchronization messages exchanged by these kinds of MAC protocols are used to learn neighborhood information and to setup a routing backbone based on that information. Thus, additional control traffic can be avoided. Moreover, nodes that are not required for routing can temporarily turn off their radio to save extra energy. The integration of routing on the MAC layer supports rather static networks. More dynamic networks such as mobile sensor networks cannot be supported on the MAC layer. Therefore, we have implemented an additional routing and topology control mechanism on the network layer that supports node mobility (see the mobility support module in Figure1.1).

Properties of backbone setup mechanisms in dependance of the used layer of the network stack are shown in Table 5.1. On the MAC layer backbone setup and maintenance information is piggy-backed on SYNC messages. Thus, no additional control messages are needed. On the other hand, this implies delays, since always the transmission of a SYNC message must be awaited to signal some control infor-

**Table 5.1:** Backbone construction on MAC and networking layer.

| | MAC | network |
|---|---|---|
| Control message overhead | minimal | high |
| Mobility support and repair | no | yes |

mation. These delays can be tolerated in static networks, where topology changes occur infrequently. To support mobile networks, an additional algorithm on the networking layer is required. Due to the usage of specific control messages this algorithm does not face the problems of piggy-backing the control information on SYNC messages. It introduces overhead, though.

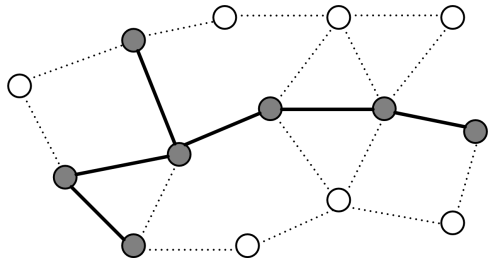


**Figure 5.1:** Routing backbone in a sensor network.

Routing and topology control are provided by a virtual backbone that is implemented as a connected dominating set (CDS) (see Section 3.3.3). An example of a backbone in a sensor network is depicted in Figure 5.1. Backbone nodes are colored gray, while non-backbone nodes are colored white. Communication links in the backbone are illustrated by the bold lines, while other communication links are indicated by dashed lines. Every non-backbone node in Figure 5.1 is adjacent, i.e., it possesses a communication link, to some node in the backbone. Hence, the network is connected and routing can be performed over the backbone. Non-backbone nodes shut down their radios and go to sleep for a long-sleep period that lasts for multiple listen/sleep cycles (see also Chapter 4).

After any long-sleep period the backbone is reestablished by the base station, taking current network conditions and energy distributions into account. Nodes with high battery levels are favored for election into the backbone. Thus, network lifetime can be extended. Independent of their state, all sensor nodes turn on their radios to organize themselves into event observing and tracking groups upon observance of an event. The according group organization algorithm (DELTA) is presented in Chapter 6. Only the leader node uses the backbone to route its event reports to the base station. Having finished their monitoring tasks, all nodes resume their assigned role in medium access and routing.

## 5.2 Routing Backbone on the MAC Layer

In this section we propose a routing backbone construction mechanism that exploits and uses the synchronization messages exchanged by synchronized contention-based MAC protocols such as T-MAC (see Section 3.2.1) or DW-MAC (see Section 3.2.1). Thus, no additional control traffic is required for routing. The routing backbone is useful for rather static applications with source-to-sink communication. By using the SYNC messages, the routing backbone extends the lifetime of the network. The base station is always rooting the backbone.

The SYNC messages intrinsically provide all network nodes with neighborhood information. This information is used to construct the routing backbone without additional control traffic. Moreover, nodes that are not required for routing can turn off their radio and can go to sleep for a long-sleep period that lasts for multiple listen/sleep periods. In this long sleep period non-backbone nodes only wake up if they have some sensor readings to be reported. If this is the case, the respective non-backbone node wakes up, synchronizes to a node in the backbone, sends its sensor readings to the backbone node and goes back to sleep again. Henceforward, the backbone node is responsible to route the sensor readings of the non-backbone node to the base station (over the backbone). If a backbone node has to report some sensor readings, it does this directly.

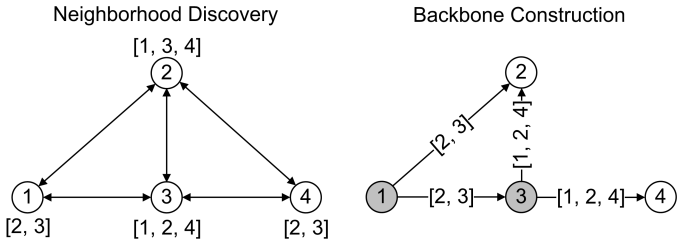



**Figure 5.2:** Neighborhood discovery and backbone construction.

Both steps to setup a backbone are depicted in Figure 5.2. Before establishing a routing backbone all network nodes learn their neighborhood from overheard SYNC messages in the neighborhood discovery period. For example, node 3 detects nodes 1, 2 and 4 as neighbors. Because not every node is able to transmit a SYNC in every listen/sleep cycle, the neighborhood discovery period covers a certain number of listen/sleep cycles. The collected neighborhood information is used in the backbone construction step to determine backbone nodes (colored gray in Figure 5.2). Again the SYNC messages are used. Elected backbone nodes (e.g., node 1) piggy-back their neighborhood information on SYNC messages. The receivers of such an extended SYNC message determine their state according to their own knowledge and the received information. In Figure 5.2 node 3 would elect itself as backbone node according to its own information and the information it has received from node 1. Different algorithms to determine backbone nodes are possible. Two algorithms based on connected dominating sets have been used.

### 5.2.1 Long Sleep on the MAC

The routing backbone has two desirable properties: First, all nodes are able to route their data over it to the base station. Accordingly, no additional network layer is required to setup connections to the base station. Second, because routing is guaranteed by the backbone, nodes that are not part of the backbone can go to sleep for a long-sleep period, which covers multiple listen/sleep cycles. The concept is illustrated in Figure 5.3.

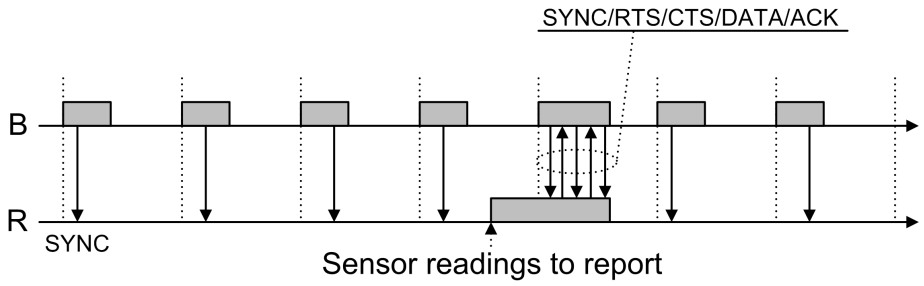



**Figure 5.3:** Operation of backbone node (B) and redundant node (R).

The operations of a backbone node B and a non-backbone node R are illustrated in Figure 5.3. B periodically synchronizes to neighboring backbone nodes, i.e., it periodically exchanges SYNC messages. R only wakes up if it has some data to be reported. It synchronizes to the backbone, allocates the channel with RTS/CTS and transmits the data (indicated by the burst of arrows). Having received a confirmation (ACK) from B, R goes back to sleep again.

In the current evaluation the learning period lasts for 10 minutes. The CDS is maintained for 50 minutes. During this time, non-backbone nodes remain asleep unless they have to report some data. Thus, non-backbone nodes save additional energy. Backbone nodes perform the normal listen/sleep cycle and do not waste more energy than running the unaltered MAC protocol. They have some information piggy-backed, but compensate for this by overhearing fewer SYNCs due to the temporarily decreased network density. Non-backbone nodes do not send SYNC messages since they are in a long-sleep state. Accordingly, our protocols save the transmission of SYNC messages compared to the standalone MAC protocol.

The batteries of nodes in the backbone have higher charge levels than the batteries of non-backbone nodes. Accordingly, the backbone is recomputed from time to time (currently every hour), taking the current battery levels of the nodes into account. In every CDS setup phase nodes with high battery levels are favored. In the following, two CDS setup algorithms based on the information collected from SYNC messages are proposed. The first algorithm needs two-hop neighborhood information, whereas the second approach requires only knowledge of the immediate neighborhood. The first algorithm is simpler and terminates faster, but requires more information.

### 5.2.2 CDS Construction Based on Multipoint Relaying

To determine the subset of next hops in the CDS, an algorithm similar to the Multipoint Relaying Protocol (MPR) (see Section 3.3.4) has been used. Any backbone node (dominator) elects its next-hop dominators such that they connect the two-hop neighbors most effectively.

#### Neighborhood Discovery

The SYNC messages are slightly modified to discover the two-hop neighborhood. In addition to the current sender $s_1$, also the sender $s_2$ of the last SYNC message received by $s_1$ is added to the SYNC message. Thus, each receiver of a SYNC message learns both, the direct neighbors ($s_1$) and the two-hop neighbors ($s_2$) over time. The reliability of the neighborhood discovery period and the needed accuracy in neighborhood knowledge are discussed in Section 5.2.4.

In addition to neighbor node $s_2$, the battery level of each sender is added to the SYNC message. The battery level is needed to account for energy distribution changes in the network. Otherwise, the nodes with best connectivity would always be chosen into the backbone, leading to fast battery depletions of these nodes.

#### Backbone Construction

Having exchanged SYNC messages for a period long enough to learn local one- and two-hop neighborhood information, the CDS process is initiated by the base station. The next-hop dominator set of a dominator node $x$ is computed as follows:

**MPR-based dominator election**

**input:** One-hop neighbor list $L_1$ and two-hop neighbor list $L_2$ of node $x$;
**ouput:** MPR set of next-hop dominators of node $x$;

**begin**
**repeat**
  **If** $\exists$ a $y \in L_1$ with exactly one link to a $z$ in $L_2$
    add $y$ to MPR;
    Remove all $z$ in $L_2$ which are now covered;
  **else**
    **For each** $y$ in $L_1$ **do**
      Compute number $\delta(y)$ of nodes in $L_2$ connected
      via $y$ that are not covered by an MPR node;
    Select the $y$ with highest product of $\delta(y)$ and battery level into the MPR set;
    Remove all $z$ in $L_2$ which are now covered;
**until** all two-hop neighbors are covered;
**end**

The CDS setup process is started by the base station (S). Each dominator elected by the algorithm is informed about its state by an extended SYNC message.

The procedure continues until all network nodes are covered. To force successful signaling, the extended SYNC messages are retransmitted either until passively confirmed by overheard SYNCs from successors or until the maximum number of retransmissions is reached.

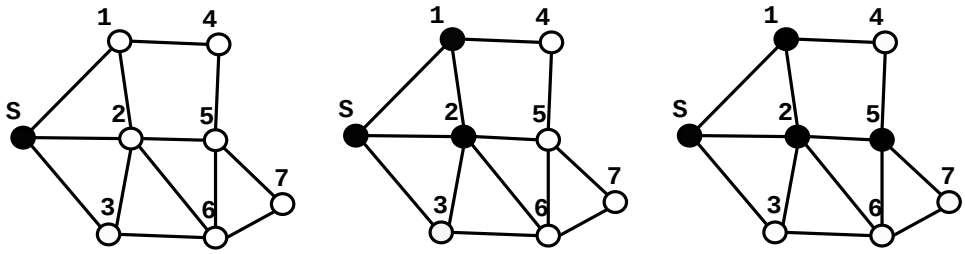



**Figure 5.4:** Example of dominator election with MPR-based CDS.

An example is shown in Figure 5.4. The base station S is the starting dominator (black). The set of next-hop dominators of S is {1, 2}, because 1 is the only node to reach two-hop neighbor 4 and 2 is the only node to reach two-hop neighbor 5. Nodes 1 and 2 cover all two-hop neighbors of S. S informs them by piggy-backing the dominator list {1, 2} to its next SYNC messages. Upon reception, both nodes become black and compute their own set of next-hop dominators. Node 1 immediately terminates its election process because no uncovered two-hop neighbors remain (nodes 3, 5 and 6 are covered by nodes S and 2). Node 2 determines node 7 as last uncovered two-hop neighbor. Because node 7 can be reached over nodes 5 and 6, step 2 of the algorithm is applied. Both nodes have the same number of uncovered neighbors, i.e., $\delta$ is 1 for both of them. Accordingly, the node with higher remaining battery level (node 5) is chosen into the CDS. The CDS setup ends when node 5 is informed about its state. All network nodes are then covered by a dominator and the algorithm terminates.

The result is obviously suboptimal. The MCDS would consist of nodes S, 2 and 5 only. However, the approximation factor is good and the main goal is to improve network lifetime rather than optimizing the CDS.

### 5.2.3 Negotiation-Based CDS

The negotiation-based CDS (N-CDS) computes the CDS without two-hop neighborhood knowledge. This time, one-hop neighborhood information is sufficient.

#### Neighborhood Discovery

The N-CDS again learns the neighborhood information from SYNC messages. This time no extension of normal SYNC messages is needed, though. Only the ID of the SYNC sender is required, which is transmitted per default. We again assume that after a given initialization period each node knows its one-hop neighbors.

### Backbone Construction

Base station S is again defined as starting node. The dominator status is determined based on a negotiation-based CDS building process that consists of four steps which are described in the following:

**N-CDS based dominator election**

1. Every dominator node broadcasts an extended SYNC message called DOMINATOR. This message contains the neighborhood list of the dominator.
2. Each receiver becomes dominated and computes a priority according to its product of battery level and number of remaining uncovered two-hop neighbors, i.e., of nodes that are not yet dominator or dominated.
3. The dominated nodes locally exchange their calculated priorities in a special SYNC message called DOMINATED.
4. The node with highest priority is elected into the backbone.

Because the DOMINATOR message contains the neighborhood list of the dominator, every receiver is informed about neighboring dominated nodes. Consider node 1 in the example in Figure 5.5. Node 1 is a neighbor of node S. Having received the DOMINATOR message from S, which contains nodes 1, 2 and 3 in the neighbor list, node 1 can determine its neighbor node 2 as dominated. The same applies to every other receiver of a DOMINATOR message. In step 3, all dominated nodes exchange DOMINATED messages containing the priority of the sender. Because each dominated node knows the dominated nodes in its one-hop neighborhood, the dominated node knows from which nodes it has to receive DOMINATED messages, in order to be able to determine the significance of its own priority (step 4). Accordingly, when a dominated node has learned all priorities from neighboring dominated nodes, it becomes a dominator if it has the highest priority among the group. If the priority of a node becomes 0, i.e., it has no uncovered neighbors left, it enters the non-backbone state after having broadcast its status in a DOMINATED message.

Due to contention or packet loss, it might happen that a dominated node cannot receive a DOMINATOR or DOMINATED message from a neighbor node in this mutual negotiation process. The node would then keep waiting for this message. To prevent this deadlock, each node sets a challenge timer. If the node does not receive any message from a specific neighbor during this period, it assigns priority 0 to the respective node. To summarize, the following terminations of the algorithm per node are possible:

1. A node determines that all its neighbors are covered. In this case the node enters non-backbone state.
2. At the moment a node determines that it has the highest priority, it enters the backbone.

3. A dominated node still has some uncovered neighbor nodes when the challenge timer expires. In this case the node enters the backbone too.

The third item is motivated as follows. Consider an uncovered node $y$ that is only connected over a dominated node $x$. In order to guarantee network connectivity, node $x$ must become a dominator. Hence, node $x$ becomes a dominator when the challenge timer expires. The challenge timer of $x$ will expire since it has an uncovered neighbor $y$. An example of the N-CDS algorithm is shown in Figure 5.5. For simplicity all nodes have the same energy level. Thus, only the node degree determines the priority of a node.

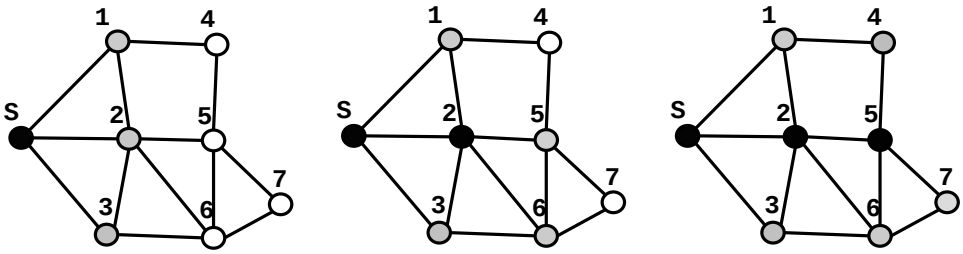



**Figure 5.5:** Example of dominator election with N-CDS.

Dominator nodes are colored black. Dominated nodes color themselves gray and uncovered nodes are white. Gateway node S starts the N-CDS algorithm by broadcasting a DOMINATOR message. Having received this message, neighbors 1, 2 and 3 color themselves gray. Node 2 computes the highest priority, because it has two neighbors (5 and 6) which are not yet covered. Nodes 1 and 3 both have only one uncovered neighbor. After having exchanged the priorities via SYNC messages, node 2 determines the highest priority (becomes black) and broadcasts the next DOMINATOR message. Receivers 5 and 6 become dominated, i.e., they color themselves gray, and exchange their priorities. Node 5 has the higher priority (neighbors 4 and 7 are not yet covered) and becomes dominator. Nodes 3, 6 and 7 go to sleep as soon as they have overheard the required DOMINATOR messages from nodes S, 2 and 5. Node 4 goes to sleep if it has overheard the DOMINATED message from node 1 and the DOMINATOR message from node 5. Node 1 finally goes to sleep if it overhears the DOMINATED message from node 4. If communication among some of the nodes was not successful, the challenge timer would expire and some of nodes 1, 3, 4, 6 or 7 might become dominators too.

Due to its negotiation character N-CDS requires a longer CDS setup time than MPR-based CDS. On the other hand, no two-hop neighborhood information is needed. The computed CDS is optimal in the example. In general, MPR-based CDS is expected to perform slightly better due to its two-hop neighborhood knowledge, though. We have developed a CDS setup mechanism that combines both approaches and provides mobility support on the networking layer (see Section 5.3). It approximates the MPR-based CDS by piggy-backing the neighbor list of

the dominator on DOMINATED messages. As N-CDS, the mechanism requires negotiation, leading to the same drawbacks.

### 5.2.4 Reliability and Backbone Reconstruction

Neighborhood discovery and backbone construction are reinitialized by the base station after every long-sleep period. Thus, the current connectivity and energy distributions in the network are considered after every long-sleep period. Local path update strategies could be an alternative. Such approaches have not been further considered due to their complexity.

The accuracy of the one-hop neighborhood information is more critical than the accuracy of the two-hop neighborhood information. Accurate two-hop neighborhood knowledge is only needed to optimize performance of the MPR-based approach. However, the two-hop neighborhood information must be complete enough to ensure that the MPR-based algorithm does not terminate without covering all network nodes. Complete coverage has been achieved in every simulation, though. Collecting neighborhood information by SYNC messages, in particular collecting two-hop neighborhood information, is a time-consuming task. Therefore, we have determined when neighborhood information must be updated. If a node depletes, the according neighborhood information of this node becomes invalid. This has impact on the backbone construction mechanisms. Two possibilities need to be distinguished:

- **Invalid one-hop neighborhood information**: Unavailable dominators could be elected. This must not happen. Hence, the one-hop neighborhood is relearned in every neighbor discovery period, i.e., before every backbone setup.
- **Invalid two-hop neighborhood information**: In the worst case a redundant dominator is elected. This is not desirable but neither critical. Hence, two-hop neighborhood information is only deleted if the relevant node has not been overheard for a certain amount of time.

In all simulations the complete one-hop neighborhood information and most two-hop neighbors were relearned in every neighbor discovery phase. Two-hop neighbors that have not been overheard were still known due to the timeout criteria.

## 5.3 Routing Backbone on the Network Layer

In this section we provide a CDS mechanism that has been implemented on the network layer and supports node mobility. Specific control messages, i.e., hello messages that are optionally extended with CDS setup information, are needed. These hello messages are used to learn neighborhood information and to setup the routing backbone. By using hello messages on the routing layer, the algorithm is decoupled from the MAC layer. Thus, node mobility can be supported, because control messages can be sent immediately. On the MAC layer, the piggy-backing of information onto the SYNC messages saves communication costs, but leads to long information dissemination delays (see Section 5.1). These delays prevent the deployment of path adaptation functionality that are needed for mobility support.

Like in the MAC-based approaches, non-backbone nodes turn their radios off and go to sleep for a certain amount of time. During these long-sleep periods non-backbone nodes periodically wake up to check network conditions. After a long sleep period the backbone is reestablished by the base station. The backbone is thus able to consider the energy load distribution over time. In addition, backbone repair mechanisms support the detection and correction of node failures and/or node mobility. The algorithm shows good approximation of an MCDS after backbone setup and is able to repair link breaks on demand with short delays and low message overhead.

The local one-hop neighborhood information is learned by periodically exchanged hello messages (beacons). Two-hop neighborhood information is only exchanged on-demand by piggy-backing one-hop neighborhood information when the CDS is established. Thus, the algorithm achieves the performance of MPR-based CDS, but requires the amount of neighborhood information of N-CDS. Like N-CDS, the algorithm requires a more complex timer handling, though.

The CDS setup information is piggy-backed on the hello messages. Thus, no additional control traffic is introduced to setup the CDS. Thereby, CDS setup or maintenance tasks interrupt the normal hello message exchange by transmitting the extended hello message prioritized. Thus, smaller latencies can be achieved. Whenever no backbone construction or repair mechanism is in action all hello messages perform their basic task. The CDS setup is again distributed. Backbone joining decisions are either based on the link degree or on a combination of link degree and remaining battery power.

### 5.3.1 Receiver-based Backbone Construction

The Receiver-based CDS (R-CDS) is an approximation of the MPR-based CDS approach (see Section 5.2.2). Instead of requiring any node to know its two-hop neighborhood, R-CDS only requires knowledge of two-hop neighborhood information on demand, though. This is achieved by piggy-backing the neighborhood information of a dominator on its hello message and distributing the information two hops into the network. The basic steps of R-CDS are listed in the following:

**R-CDS based dominator election**

1. Every dominator node $x$ broadcasts extended hello messages called DOMINATOR. These messages contain the neighborhood list of the dominator.
2. Each receiver $y$ becomes dominated and broadcasts an extended hello message called DOMINATED. This message contains again the neighborhood list of the dominator.
3. Receivers $z$ of DOMINATED messages compare their own neighborhood information with the neighborhood information of the dominator. The sender $y$ is prioritized according to this comparison:
   - 3.1. If $y$ is the only node that connects $x$ and $z$, $y$ is chosen with high priority as dominator.
   - 3.2. If there are multiple paths between $x$ and $z$, node $y$ is prioritized according to its link degree and/or its remaining battery level. The priority must be lower than in [3.1], because nodes $z$ with only one path to the dominator must be connected.
4. Node $z$ informs node $y$ about its priority. The algorithm is receiver-based.

With the described mechanism the neighborhood information of the dominator on demand is disseminated two hops into the network. Thus, every receiver of a DOMINATED message is able to decide if it is located two-hops away from the dominator. Furthermore, it can determine the number of paths to the dominator. According to this number, the sender of the DOMINATED message is prioritized. The most relevant step in the Multipoint Relaying Protocol (see Section 3.3.4) is to choose nodes as Multipoint Relays that provide exactly one path to a node two hops away. To do so, the Multipoint Relaying Protocol requires two-hop neighborhood information. In R-CDS the neighborhood information of the dominator is delivered to its two-hop neighbors. Thus, the two-hop neighbors are able to determine the number of paths to the dominator. With this mechanism, R-CDS approximates Multipoint Relaying, but requires the setup of two-hop neighborhood information only on demand.

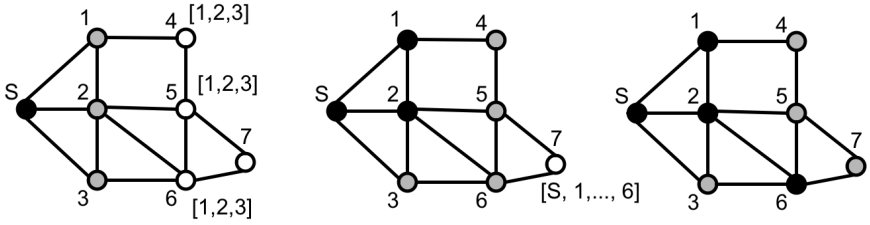



**Figure 5.6:** Example of dominator election with R-CDS.

An example of a backbone construction of R-CDS is illustrated in Figure 5.6. The base station (S) broadcasts a DOMINATOR message. Each receiver of DOM-

INATOR message (nodes 1, 2 and 3) sends a DOMINATED message. Having received the respective DOMINATED message(s), two-hop neighbors 4, 5 and 6 know the neighbor list [1, 2, 3] of S. Node 4 determines only one path to S over node 1. Therefore, node 1 is chosen as dominator. Similar, node 5 determines also only one path over 2 to S and elects 2 as dominator. Both nodes are immediately informed about their status. Node 6 delays its decision, which would also choose node 2 due to higher link degree, because it has two paths over 2 and 3 to S. Nodes 1 and 2 broadcast DOMINATOR messages. The receivers 3, 4, 5 and 6 are thus again instructed to send DOMINATED messages. By analyzing the DOMINATED messages, node 7, the only remaining uncovered node, learns the one-hop neighborhood [S, 1, 3, 4, 5, 6] of node 2. Because node 2 determines 2 paths to the base station over nodes 5 and 6, which provide both the same link degree of 4, node 7 either elects node 5 or 6. In this example it elects node 6 due the higher remaining battery level of node 6. In order to make this last decision, every dominated node attaches its battery level and its link degree to the DOMINATED message.

The algorithm terminates as soon as no uncovered nodes remain. This happens in every connected network, provided that all required messages have been successfully transmitted (see the next section). Two terminations are possible:

*(i)* Nodes adjacent to a dominator become dominated.

*(ii)* Any node $y$, two hops away from a dominator chooses an up-link node $z$ as dominator. If node $y$ is not covered by another dominator in the meantime, node $z$ will win the dominator election and inform node $y$ accordingly. Node $y$ will become dominated in both cases according to *(i)*.

### Timer Handling and Reliability

As indicated above the control messages used for the backbone construction are included in the hello messages. As soon as a node enters backbone construction state it sets a CDS control message retransmission counter to cover three hello intervals and includes its current control info in its hello messages during that time. Accordingly, the respective control information is transmitted three times. With this mechanism the algorithm accounts for possible packet loss. The retransmissions impose only minor overhead, because the hello messages, which would be exchanged anyway, only have to be extended with some information. Consequently, only the packet size is temporarily increased, but no additional control messages are generated. To decrease CDS setup time and to support the prioritization of nodes, the extended hello messages are transmitted in a prioritized manner.

To determine the release time of the extended hello messages two intervals are defined: a short interval of 100 ms and a long interval of 1 s. The intervals are chosen according to typical sensor network properties. In our real-world implementation of the R-CDS algorithm (see Section 5.4.3) we have used the ESB sensor node platform (see Section 3.8.1). ESB sensor nodes need 14 ms to switch

to send mode, send the packet and switch back to idle mode. Therefore, a contention period of 100 ms for the short interval is reasonable. The long interval is needed to delay dominator elections if multiple paths to the last dominator exist. The timers for the DOMINATOR and the DOMINATED messages are both randomly chosen from the short interval. The timer for the dominator choice message sent by a two-hop neighbor is chosen from both intervals depending on the priority of the backbone candidate. If only one path to the last dominator exists, the timer is chosen from the short interval. If multiple paths to the last dominator exist, the long interval is also considered. As soon as a control message timer is set, the periodic hello sending mechanism is interrupted and the next hello is scheduled according to the control message timer. In sensor networks algorithms are confronted with high packet loss due to collisions or bit error rates and small bandwidth. An integration of the control messages and the hello messages is therefore helpful as it scales down the control traffic load and increases the probability of message delivery, while avoiding additional RTS/CTS like mechanisms.

The backbone is maintained for a predefined backbone time. After this time, the backbone is reestablished, adapting itself to the new network conditions. During backbone time the dominated nodes follow a listen/sleep schedule. The dominated nodes wake up periodically and listen to the medium before going to sleep again. If a dominated node detects a link break in its vicinity or if it did not sense any dominator at all during its listen period, the node stays awake. The details of the backbone repair mechanisms are explained in the next section.

### Local Path Adaptation and Repair

In order to deal with dynamic network topologies a local path adaptation and repair mechanism has been implemented. Two different kinds of link breaks might evolve:

*(i)* a backbone node determines an up-stream link break.

*(ii)* a dominated node detects its isolation from the backbone.

In both cases a link break is detected if a node did not overhear any hello message for a certain amount of time, i.e., three times for the hello interval in *(i)*, and for the duration of the listen period in *(ii)*. In the latter case the dominated node remains awake and tries to connect to a dominator. In both cases the link break detecting node enters link-break state and starts to broadcast link-break notifications to inform its neighborhood about the link break. Each node overhearing a link-break notification enters link-break state too. It saves the address of the node announcing the link break and starts to propagate the link-break information further. As soon as a backbone node with a valid route to the base station (BS) receives a link-break message, it enters the path-update state and broadcasts its own path to the BS within its next hello message. Nodes with valid routes to the BS neither detected a link break themselves nor were they informed about a link break.

Each node in the link-break state overhearing such a path update message adopts the path and rebroadcasts the updated path (including its own ID). If the rebroadcasting node is a dominated node it enters the backbone and becomes dominator. In general not all dominated nodes in the path repair state are needed to repair a path. To keep the number of resulting dominators small, the path update distribution is done under contention. Thus, dominated nodes that detect nodes in repair state renounce their own path repair functionality. As mentioned above all affected nodes setup a list of the nodes they received link-break messages from. Accordingly, all nodes overhearing a path-update message from a node in their list of link break reporting nodes, cancel their own path-update procedure.

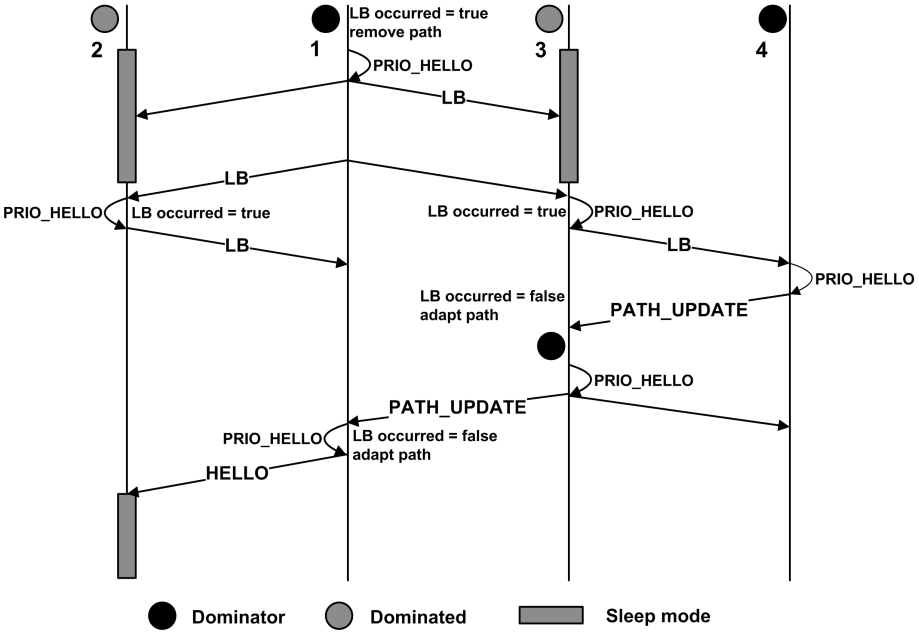



**Figure 5.7:** Link break repair mechanism.

The message flow to repair a link break that has occurred at dominator 1 is depicted in Figure 5.7. Dominator 1 detects a link break to its up-link dominator node (not depicted) and broadcasts a link-break message accordingly. As soon as there are some neighbors awake (nodes 2 and 3) they enter link-break state and propagate the link break further. As soon as dominator 4 with a valid route to the BS overhears an link break message, it responds with a path update message. Dominated node 3 overhears this message, adapts the path, becomes a dominator and finally broadcasts the path update message extended with its own ID. Having overheard the path update message from the new dominator 3, node 1 knows that the path is repaired, adopts the new path to the base station and resumes its normal functionality. The next hello message sent by 1 informs node 2 that the link break has been repaired. Accordingly, node 2 can go back to sleep again. In Figure 5.7 only the relevant transmissions are shown.

## 5.4 Evaluation

So far, backbone construction mechanisms on the MAC and on the networking layer have been proposed. The approaches on the MAC layer have an intrinsic advantage in energy saving. On the other hand, the backbone construction mechanism on the networking layer supports mobility. Due to their different goals and implementations a direct comparison of the algorithms is not meaningful. Therefore, the approaches have been evaluated separately starting on the MAC layer.

### 5.4.1 Simulations on the MAC layer

MPR-based CDS and N-CDS have been evaluated in simulations. T-MAC has been chosen as underlying MAC protocol. For compatibility and comparison reasons T-MAC and an optimal pre-configured shortest-path routing tree have been implemented according to [142]. Any implementation of a routing algorithm on top of T-MAC has a different impact depending on the selected routing protocol and is thus difficult to be evaluated. Therefore, we have decided to compare our approaches to optimal benchmarks.

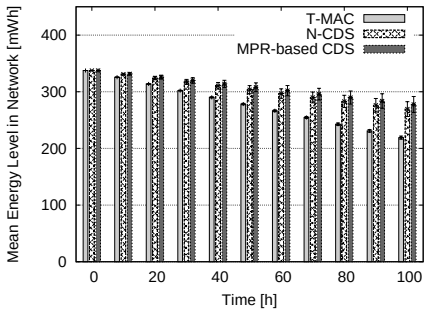
#### Simulation Scenario and Parameters

The CDS is rebuilt every hour. The long-sleep period lasts 50 minutes and the learning period 10 minutes. Nodes in long-sleep state quickly wake up every minute to send their sensor readings to the base station (see also Figure 5.3). Having successfully synchronized to the backbone and transmitted their data, they go back to sleep. The network parameters are similar to those used in the simulations of LACAS. The parameters can be found in Table 4.1. The properties of the sensor nodes are again configured according to values from the Embedded Sensor Board (ESB) platform (see Section 3.8.1).

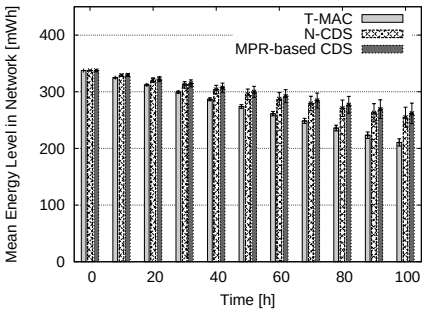
#### Average Energy Consumption in the Network

In a first evaluation the average remaining energy level of the batteries in the network has been logged for each protocol. Each node has initially been charged with 335 mWh. 100 hours have been simulated. It has been ensured that all nodes have enough energy so that they do not run out of energy during simulation time. Node failures were not in the scope of the current evaluation.

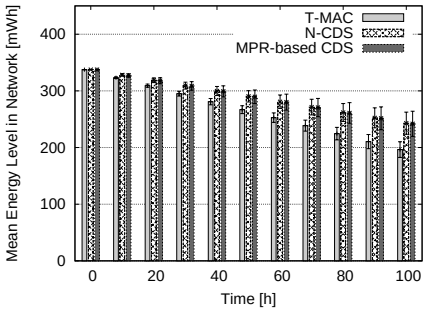
Figure 5.8 shows the average remaining energy level for all tested protocols and network sizes. Both N-CDS and MPR-based CDS perform better than T-MAC. As expected, MPR-based CDS is slightly more efficient than N-CDS. The results show that the energy savings by non-backbone nodes compensate for the piggybacking of control data. The network size has some impact on the on average consumed energy in the network. This increase in energy consumption is however due to the higher data traffic load caused by the increased number of reporting sensor nodes (every node transmits a data message every second). The backbone




**(a)** Network size of 50 nodes.




**(b)** Network size of 100 nodes.



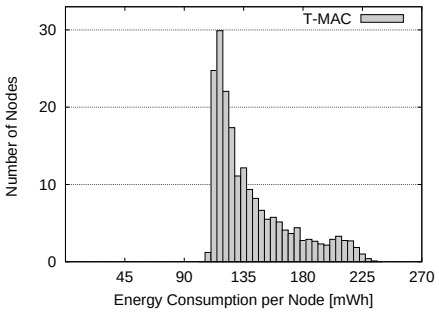

**(c)** Network size of 200 nodes.

**Figure 5.8:** Average remaining energy in the network.

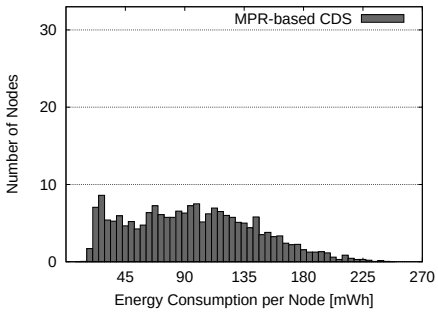
performance is not affected by the network size. In our evaluation T-MAC has been used. Because SYNC messages are used, T-MAC could be substituted by other synchronized MAC protocols. The benefits of the backbone should be the same. Because T-MAC has been provided with cost-free routing, implementing routing on top of T-MAC would additionally strain energy consumption of T-MAC.

### Energy Distribution in the Network

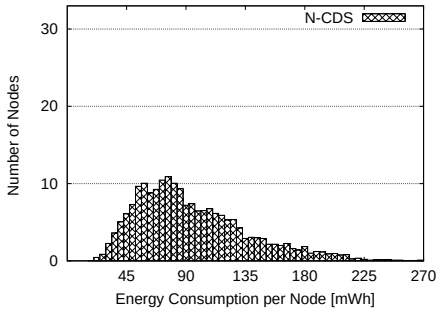
The goals of our work have been to provide routing on the MAC layer to save energy, and to distribute the energy consumption uniformly over the network. This uniform distribution of the energy consumption is more meaningful to extend network lifetime than the in-average consumed energy, because the probability that nodes discharge quickly is decreased, which is in particular critical if irreplaceable nodes are affected. Figure 5.9 shows the energy distribution of the different algorithms in the network consisting of 200 nodes (Figure 5.8(c) shows the average energy consumption). The results for networks of 50 and 100 nodes are similar.




**(a)** T-MAC.




**(b)** N-CDS.




**(c)** MPR-based CDS.

**Figure 5.9:** Number of nodes with a specific energy consumption.

The distribution of nodes with similar energy consumption differs significantly between T-MAC and the CDS-based approaches. In T-MAC all nodes consume more than 100 mWh in 100 hours simulation time and the percentage of nodes with high energy consumption is higher than that of the CDS-based approaches. In T-MAC there are numerous nodes (about 30) with an energy-consumption of about 110 mAh (see Figure 5.9(a)). The other nodes consume even more energy. On the other hand, the CDS-based approaches charge many nodes less.

The high fraction of nodes with low energy consumption in CDS-based approaches, i.e., the nodes on the left side in Figure 5.9(b) and 5.9(c), indicates that the periodic setup of the CDS leads to a good energy consumption distribution. However, there are also nodes with high energy consumption. These are the nodes on the right side in Figure 5.9(b) and 5.9(c). Examining the different solutions, neither N-CDS nor MPR-based CDS increase the number of heavily strained nodes compared to T-MAC. Accordingly, the CDS-based algorithms do not increase the number of quickly depleting nodes. The existence of such nodes cannot be avoided, though. The reason is that such nodes are often or always elected into the backbone,

because there are no or few alternatives to route information to the base station. To circumvent this drawback, the node distribution in the network would have to be considered during deployment.

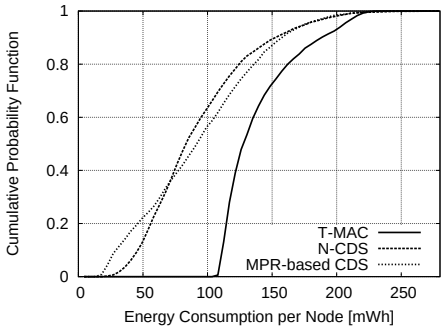



**Figure 5.10:** Cumulative distribution function of energy consumption per node.

The cumulative distribution function ($cdf$) of nodes with a specific energy consumption is shown in Figure 5.10. This function shows the percentage of nodes that consume a certain amount of energy, and it is used to assess the distributions in Figure 5.9. MPR-based CDS and N-CDS both outperform T-MAC. Figure 5.10 shows that with MPR-based CDS and N-CDS approximately 60% of all nodes consume less than 100 mWh during the simulation time of 100 hours. On the other hand, in T-MAC every node consumes more energy during the same time. It is not possible to determine whether MPR-based CDS or N-CDS performs better.

To conclude, CDS-based algorithms not only consume less energy on average (see Figure 5.8(c)), but also lead to a better energy consumption distribution. The probability of network partitions is higher for T-MAC than for the CDS approaches due to the higher number of heavily strained nodes. Since SYNC messages are used, T-MAC could be substituted by other synchronized MAC protocols (e.g., by DW-MAC). The backbone construction would remain the same.

### Packet Loss

Finally, the average data packet loss of the different protocols has been investigated. Only data transmissions have been considered.

The results are depicted in Figure 5.11. The optimal routing implemented on top of T-MAC leads to very good performance. T-MAC introduces almost no packet loss in all simulations. On the other hand, both N-CDS and MPR-based CDS lead to packet loss of up to 4%, which is still reasonable. Implementing a routing protocol on top of T-MAC might increase packet loss too due to additional signaling and suboptimal routing decisions. Nevertheless, T-MAC could still perform better than the CDS approaches, because the CDS mechanisms select many backbone nodes with poor link quality.

The reason is depicted in Figure 5.12. Since a remote neighbor $y$ of $x$ has a

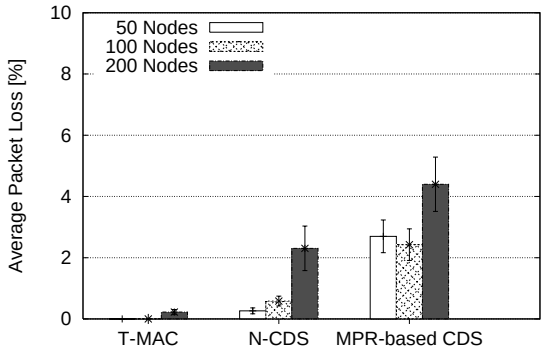



**Figure 5.11:** Average packet loss.

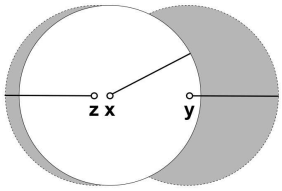



**Figure 5.12:** Area of additional coverage of neighbor nodes of $x$.

larger area of additional coverage (colored gray) than a nearby neighbor $z$, $y$ in general connects more two-hop neighbors of $x$ than $z$. On the other hand, remote neighbors $y$ have a comparatively low Received Signal Strength Indicator (RSSI) and are more exposed to interferences. Accordingly, these nodes have worse link quality than nearby neighbors.

The election procedures of both CDS-based approaches prefer neighbors that have a high probability to reaching additional nodes, because both CDS-based algorithms aim at optimizing additional coverage. However, exactly these nodes are the nodes with high probability of poor link quality, because they are located at the border of the parent dominator node. The impact could be lowered by monitoring the RSSI of incoming messages at the nodes. Thus, links with poor quality could be avoided in the backbone election procedures. In addition, symmetric links would be reinforced too.

### 5.4.2 Simulations on the Network Layer

R-CDS has been evaluated either considering link degree (R-CDS-LD) or considering the product of link degree and remaining energy level of a node (R-CDS-E). In order to model mobility we have used the Mobility Framework [100]. The properties of the sensor nodes are again configured according to values from the Embedded Sensor Board (ESB) platform (see Section 3.8.1). Other important simulation parameters are listed in Table 5.2.

**Table 5.2:** Parameters used in the R-CDS simulations.

| Parameter | Value |
|---|---|
| Network density | {12, 20} neighbors |
| Node mobility (constant speed) | {0.1, 0.5} $\frac{m}{s}$ |
| Network size | {50, 100, 200, 400} |
| Node deployment | random, but guaranteeing connectivity |
| Simulation duration | 10 h |
| Backbone period | 30 min |
| Active period | 30 s |
| Sleep period | {2, 4, 8} · 30 s |

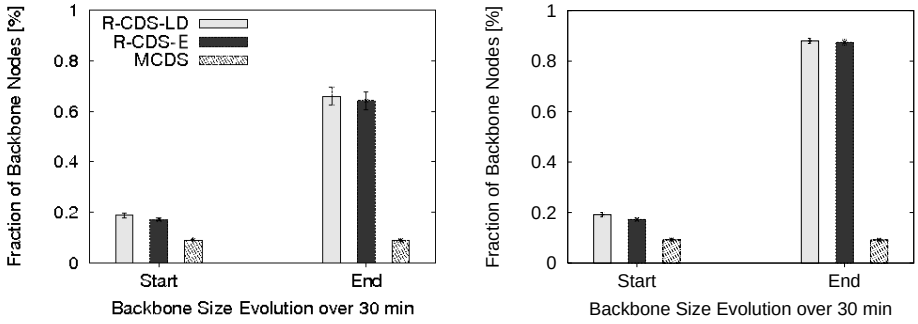
We have simulated sparse networks with approximately 12 neighbors per node and dense networks with 20 neighbors per node. All nodes permanently move at a common constant speed of either $0.1\frac{m}{s}$ (low mobility) or $0.5\frac{m}{s}$ (high mobility). Network sizes of 50, 100, 200 and 400 nodes have been simulated. We did not simulate static networks, because no link breaks would occur. Twenty backbone periods have been simulated in each run (10 h / 30 min). Non-backbone nodes are periodically active for 30 s. Then they go back to sleep. Each experiment has been tested with ten seed values. The 95% confidence intervals have been computed. The confidence intervals are only shown if they are relevant. Only the simulation results of the simulations with 200 nodes are shown. Simulations with the other network sizes showed similar performance.

In the evaluation we show that both R-CDS-LD and R-CDS-E minimize the number of nodes in the backbone well, i.e., they approximate the MCDS well. We furthermore show that the fraction of link breaks that cannot be repaired is small and that the link breaks are repaired fast. Moreover, we show that R-CDS-E is at least as good as R-CDS-LD concerning the average remaining energy level of the nodes. Finally, we show that R-CDS-E shows smaller energy level variations than R-CDS-LD, which means that the network load is better balanced and that the overall network lifetime can thus be extended.

### Size of the Backbone

In Figure 5.13 the backbone sizes of R-CDS-LD and R-CDS-E as well as the backbone size of the MCDS of the networks, i.e., of the optimum, are depicted. In every simulation run, 20 backbone periods have occurred. Figure 5.13 show the average backbone size evolution over all backbone periods in all simulations. The results show that the MCDS remains nearly constant in every backbone period.

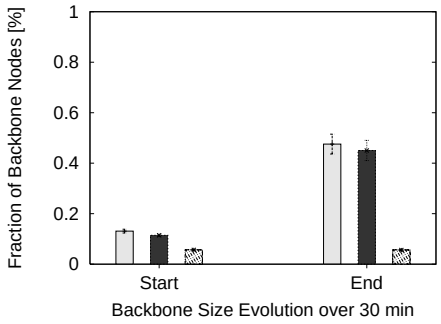
Both CDS-LD and CDS-P approximate the MCDS quite well at the beginning (start) of a backbone period, i.e. shortly after the backbone has been built. The high percentage of nodes in the backbone at the end of a backbone period (end) is



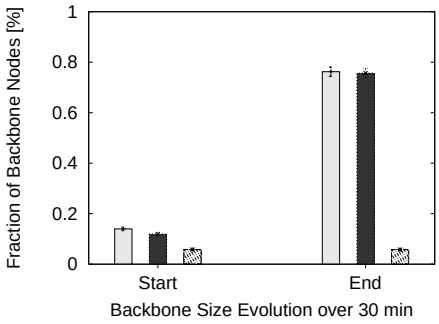

**(a)** Sparse - low mobility




**(b)** Sparse - high mobility




**(c)** Dense - low mobility



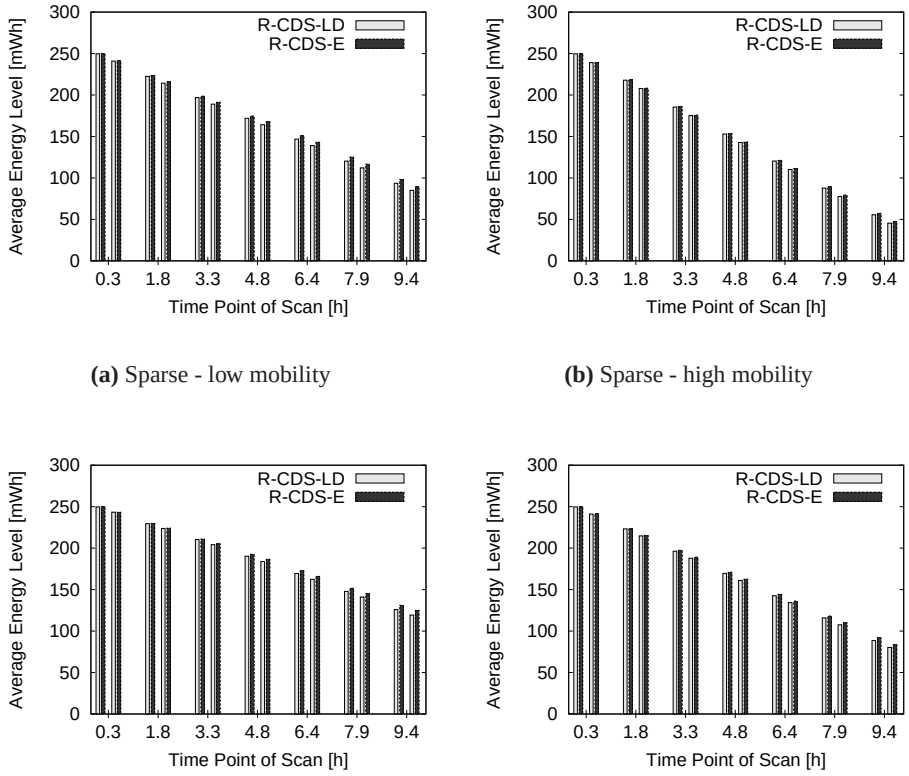

**(d)** Dense - high mobility

**Figure 5.13:** Backbone size evolution for R-CDS-LD/P and MCDS.

linear to the number of link breaks that have occurred. For each link break, nodes which repair that link break are needed. All these nodes enter the backbone. In the current version of the algorithm no mechanism to give up the dominator state has been foreseen. Thus, an increase in the number of dominators during the backbone period is obvious. Figure 5.13 shows that the evolving backbone sizes are smaller in networks with low mobility, where fewer link breaks occur. The same applies for network density (see Figures 5.13(c) and 5.13(d)). This is of course not surprising. The backbone size would even remain constant in a static network. Considering mobility, the increasing number of dominators is acceptable because link breaks are repaired. Thus, network connectivity can be guaranteed and correct paths to the base station are supported. Five and more link breaks occurred in the current evaluation per long-sleep period, even in the dense and low mobility scenarios. In static networks such a high number of node failures in half an hour would be rather unrealistic.

### Average Energy Consumption of the Network Nodes

In this section we present the results of average energy consumption in the network over time (see Figure 5.14). Both R-CDS-LD and R-CDS-E show linear charging of batteries and almost the same battery level in all simulations.




**(a)** Sparse - low mobility

**(b)** Sparse - high mobility

**(c)** Dense - low mobility

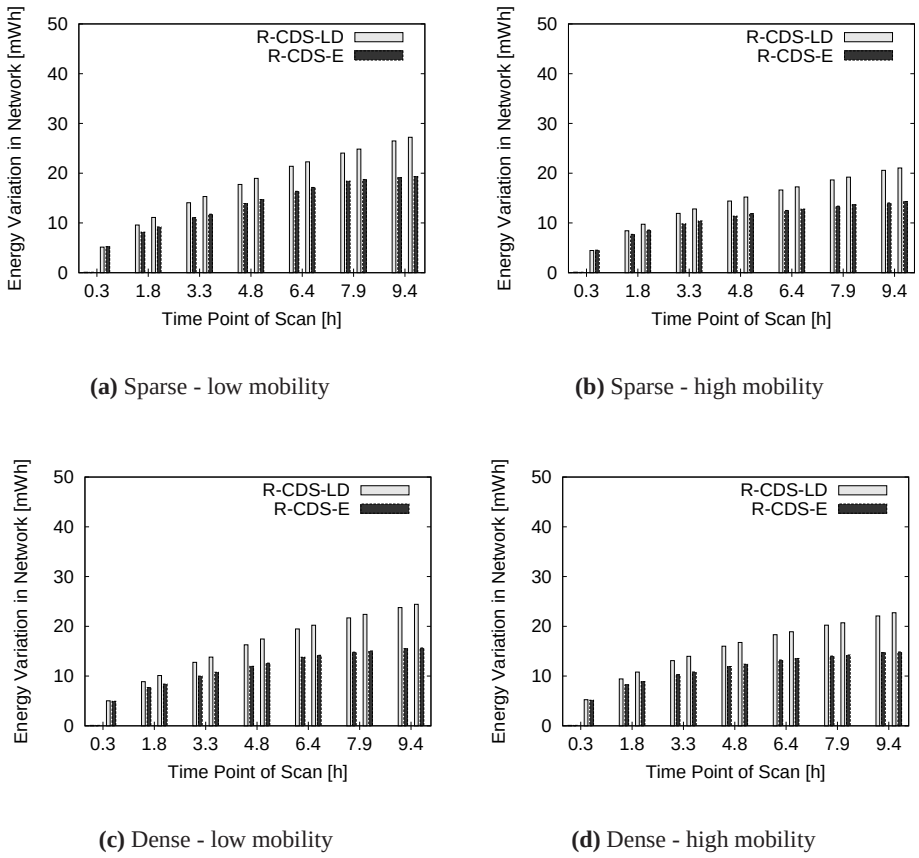
**(d)** Dense - high mobility

**Figure 5.14:** Average remaining battery level in the network for R-CDS-LD/P.

Consequently, in terms of average energy consumption neither R-CDS-LD nor R-CDE-P shows a significant advantage. In sparse networks with high mobility more link breaks occur. This leads to more dominators and consequently also to less dominated nodes that are able to go to sleep. Accordingly, the batteries are more drained. On the other hand, in dense networks with low node mobility the average remaining battery level is higher. Obviously, performance increases, the denser and more static a network is.

### Distribution of the Energy Consumption over the Network Nodes

In contrast to the average battery level, which is almost the same for both R-CDS-LD and R-CDS-E, the distribution of the battery levels over the nodes in the network are considerably smaller for R-CDS-E than for R-CDS-LD (see Figure 5.15). Every network node has the same battery level of 55 mAh in the beginning of the simulation. Accordingly, the variation in the battery levels is small in the beginning and increases with time.




**(a)** Sparse - low mobility

**(b)** Sparse - high mobility

**(c)** Dense - low mobility

**(d)** Dense - high mobility

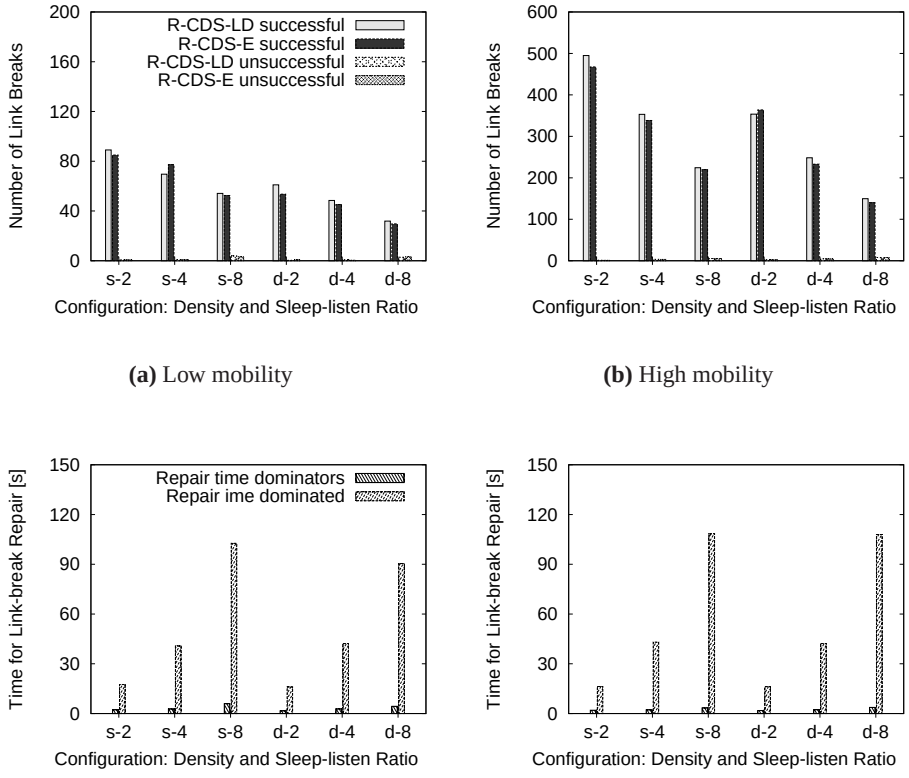
**Figure 5.15:** Variation of energy consumption among the network nodes for R-CDS-LD/P.

A balanced energy consumption distribution in the network indicates a uniform charging of the batteries amongst the network nodes. On average this will extend the network lifetime, because fewer nodes might run out of energy quickly. Thus, network partitions are prevented. With R-CDS-LD a certain fraction of nodes (those with highest node degrees) are almost always elected into the backbone. Accordingly, these nodes will run out of battery soon. In addition to the depletion of (critical) nodes, the resulting smaller node density has a negative impact on the average battery level (see also the last section). Thus, the probability of network

partitions is increased. On the other hand, R-CDS-E considers the current battery level when electing dominator nodes. Thus, a more uniform energy consumption load can be achieved in the network. Of course, performance increases if more redundant nodes are available, because more dominator candidates are available for substitution. In our simulations only the first ten hours of operation of a network have been simulated. The ratio between R-CDS-LD and R-CDS-E would increase over operation time.

### Number of Link Breaks and Repair Time

Finally, the number of successful and unsuccessful link-break repairs as well as the average time to repair a link break has been evaluated. The number of successful and unsuccessful link-break repairs are shown in the upper part in Figure 5.16, whereas link-break repair times are shown in the lower part (s = sparse, d = dense; 2,4 and 8 are the sleep-listen ratios).




**(a)** Low mobility **(b)** High mobility

**(c)** Low mobility **(d)** High mobility

**Figure 5.16:** Link break repair performance for R-CDS-LD/P.

Figure 5.16 shows that the number of unsuccessful link repairs is small in all

simulations, while the number of link breaks increases considerably with higher node mobility. The average time to repair a link break that occurred at a dominator node was below 5 seconds in all simulations. On the other hand, the average time to repair a link break that occurred at a dominated node is considerably longer and increases with higher sleep-listen ratios, which is not surprising. A dominated node $x$ that detects no connection to the backbone has to wait until another dominated node $y$ wakes up. Then, $x$ has to inform $y$ about the link break, before $y$ searches for a dominator node itself. This introduces long repair delays that increase linearly with the sleep time of the involved nodes.

### 5.4.3 Real-World Experiments

R-CDS-E (considering energy level) has been implemented on the ESB sensor nodes (see Section 3.8.1). The basic functionality is the same as in the simulations. However, the required intervals have been adapted to consider properties of the ESB nodes in a real deployment. The values are listed in Table 5.3.

**Table 5.3:** Duration / periodicity of the different relevant intervals.

| Interval / Timer | Duration [s] |
|---|---|
| HELLO | 2 |
| LISTEN | 6 |
| SLEEP | 10 |
| LONG SLEEP | 1800 |
| CONTENTION WINDOW | 0.2 |
| RETRANSMISSION DELAY | 1 |

All networks are static. First, the resulting backbone size of R-CDS-E is evaluated. Second, we tested the repair mechanism by turning off one of the dominator nodes. The CDS setup has been repeated 10 times. Before the CDS algorithm started, four HELLO messages were exchanged to setup the neighbor tables on each node. Each control message, i.e., each HELLO message extended with CDS relevant information, is retransmitted five times to ensure reliability. The experiments are described and discussed in the following.

The network topology for the CDS backbone size experiment is depicted in Figure 5.17. The MCDS of the network is indicated as gray sensor nodes in Figure 5.17. In this experiment the size of the resulting CDS and the duration of the CDS setup phase have been determined.

The times that elapsed until the last CDS setup control messages were sent, were between 14.7 and 17 s. On average the backbone setup lasted 15.6 s until the last control message was sent. This seems to be rather long. However, it can be explained as follows: The CDS setup process starts when the first HELLO packet is sent. As mentioned above, four HELLO messages have to be exchanged

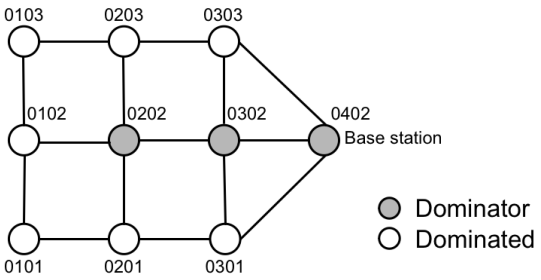



**Figure 5.17:** Network topology and MCDS of the experiment.

to setup the neighbor lists. This results in $4 \cdot 2s = 8s$. The DOMINATOR and DOMINATED messages sent by the nodes are delayed on average for 100 ms (see Table 5.3). Each retransmission introduces an additional delay of another second. Accordingly, on average $4 \cdot (1000ms + 100ms) + 100ms = 4.5s$ are spent to support reliable transmission of a control message. Obviously, the last transmitted control message also requires these 4.5 s. The actual CDS setup time is therefore the average backbone setup time of 15.6 s minus the 12.5 s introduced by the neighborhood learning and reliability support functionality, i.e., 3.1 s, which is easily tolerable considering a backbone period of half an hour.

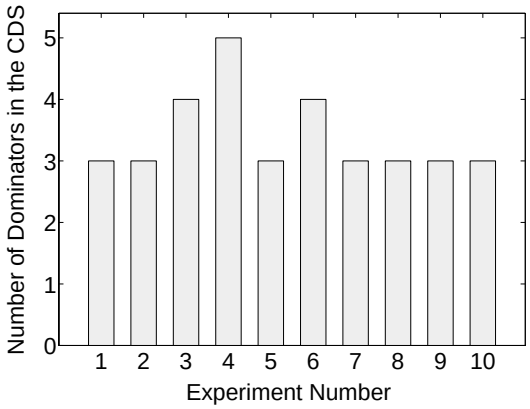



**Figure 5.18:** Number of dominators in the experiments.

On average 3.4 nodes have evolved as dominators in the experiments. The results are shown in Figure 5.18. In the worst case five nodes have been elected into the backbone, whereas in 70% of the CDS establishments the MCDS has been achieved. Such a good approximation of the MCDS is due to the small network size. In larger networks the MCDS could barely be achieved. For the case with five dominators, an analysis of the network traffic has shown that some nodes did not overhear each other, which is a well-known problem in wireless sensor networks [175].

In addition we have analyzed the median distance in hops from the nodes to the BS. According to the topology, nodes 101, 102 and 103 are three hops away from the BS, whereas nodes 201, 202 and 203 have a distance of two hops and nodes 301, 302 and 303 are adjacent to the BS. In every computed CDS, nodes 101, 102 and 103 are indeed connected to the BS over three hops. However, the second triple (nodes 201, 202 and 203) is in reality on average 2.7 hops away from the BS instead of the optimum of 2 hops, and even the three nodes adjacent to the BS have an average hop distance of 1.9 hops. This is again due to the communication holes caused by radio irregularities and temporal radio failures.

In order to evaluate the link repair functionality in a static real network, we have setup a tailored network topology and have turned off a node. An alternative path existed. The experiment setup is depicted in Figure 5.19.

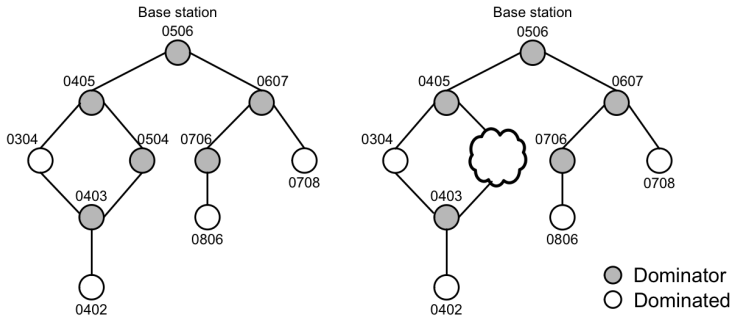



**Figure 5.19:** Link repair experiment. Node 504 is manually turned off

After the successful setup of the CDS, either node 304 or 504 is manually turned off, depending on which one has been elected into the CDS. In Figure 5.19 it is node 504. The link repair results are shown in Figure 5.20.

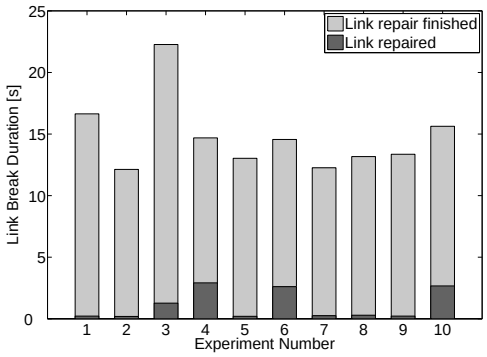



**Figure 5.20:** Link break repair time in the different CDS cycles.

In every experiment the link break has been repaired within 3 seconds. The variation is rather high, ranging from 200 ms to 2.7 s. This can be explained by

the state of the nodes involved. If all nodes are awake, the link break messages are forwarded quickly and the link failure is accordingly quickly fixed. On the other hand, if the dominated node (304 or 504) is sleeping while the link break occurs, the repair time is delayed until the dominated node wakes up. Depending on the duration of the listen period and the time point of node failure, a link break could remain undetected for an unpredictable amount of time, ranging from considerably less than a second to a couple of seconds. In addition to the actual link repair time, the time until the last control message reporting about the current link break has been sent is shown. The average time until the last control message has been sent is 14.7 s. Due to retransmissions and the dissemination of the link break, this period obviously lasts much longer than the actual time to repair the link break. However, this signaling of the link break state only imposes some additional information that has to be transmitted in the hello messages, but does not affect CDS performance. In every case the link break messages have been kept on the left branch of the network. Accordingly, the right branch has never been involved in the link repair.

## 5.5 Conclusions

In this chapter the construction of a routing backbone to support our event detection framework with routing and medium access has been discussed. The maintenance of a routing backbone allows the temporary disconnection of non-backbone nodes, because these nodes are not required for routing. Thus, additional energy can be saved. The backbone construction has either been implemented on the MAC layer or on the networking layer:

- MAC layer: The SYNC messages exchanged by the network nodes have been used to setup a (routing) backbone on the MAC layer. Non-backbone nodes have been temporarily sent to sleep for multiple listen/sleep cycles if no events have been present. It has been shown that the proposed algorithms (MPR-based CDS and N-CDS) save additional energy on the MAC layer, while routing is intrinsically supported. No additional signaling for routing is required. To account for changing battery levels of the nodes the backbone is periodically reestablished. This further extends network lifetime. Data throughput was slightly affected by the virtual backbones. The increase in packet loss depends on poor links, which are established in the CDS setup phase. The impact has been identified and could be decreased by monitoring and avoiding links with poor link quality. The drawback of this algorithm is a rather long control message dissemination delay. Therefore, the approach fails if nodes are mobile.

- Network layer: R-CDS has been implemented on the network layer. Due to the decoupling of the CDS control messages from MAC functionality, CDS setup and maintenance relevant information can be disseminated faster. Thus, node mobility can be supported. Two versions of R-CDS have been

evaluated in simulations. The versions are either based on the link-degree of nodes (R-CDS-LD) or on the product of link degree and remaining node energy (R-CDS-E). Both CDS-LD and CDS-P perform efficiently and show good approximations of the MCDS of the network in simulations. Link breaks are repaired quickly with both versions. R-CDS-E outperforms R-CDS-LD in terms of energy consumption variations, which makes it a better choice to extend network lifetime. Therefore, R-CDS-E has been chosen for implementation on real sensor hardware. The evaluation has shown that the R-CDS-E algorithm approximates the MCDS well in real-world. Also link breaks have been repaired quickly in real world experiments.

All approaches have shown good performance in simulations. The MAC layer approaches have not been compared to R-CDS. The different goals of the algorithms don't make a direct comparison meaningful. R-CDS provides mobility support, while N-CDS and MPR-based CDS provide medium access. Only the routing is common to all three approaches. The backbone election procedure of R-CDS could be replaced by the functionality of N-CDS or MPR-based CDS or vice versa. Mainly, the kind of used messages (HELLO or SYNC) would have to be adapted. Due to the affinity of MPR-based CDS and N-CDS algorithms to R-CDS, similar performance could be expected.

In the remainder of the thesis the lower layers of the networking stack are left and the application-layer features of the event monitoring system are presented. The next chapter introduces the event detection and tracking group formation functionality.

Chapter 6

# Distributed Event Detection and Tracking

The previous chapters have addressed medium access and routing, which are required by our system to perform communication and reporting. In this chapter we start with the first application-specific task of our event detection system (see also Figure 1.1). In this chapter the detection and tracking of suddenly appearing events is addressed. The Distributed Event Localization and Tracking Algorithm (DELTA) is introduced. The chapter is based on work published in [98], [156] and [150]. Upon appearance of events, sensor nodes located in the event sensing area organize themselves dynamically into tracking groups. A tracking group is an organized group of sensor nodes that are temporarily responsible to monitor, track and report an event. In DELTA, the dedicated group leader informs its group members about its leader state, requests event-relevant data from its members and performs a leader handover if the event moves away. Subsequent classification of the event is performed based on the collected event information.

## 6.1 Introduction

The detection and tracking of events has gained much attention in wireless sensor network research. The related work has been presented in Section 3.4. Even though some related work provides useful functionality, current detection and tracking methods cannot be adapted to our system. They either impose too high communication load or they do not offer the accuracy required by our system. Therefore, we have developed our own solution to support the diverse requirements of an accurate long-living event monitoring system. Considering detection and tracking, our system aims at optimizing the trade-off between communication cost savings and providing the required accuracy. In order to meet the required accuracy, sufficient data needs to be collected. Accordingly, DELTA optimizes communication load to provide a requested accuracy with minimum overhead. This means, in addition to the basic detection and tracking functionality, DELTA collects information to perform localization and classification tasks. In related work these tasks are of-

ten individually addressed. The approaches either focus on efficient detection and tracking, or on collecting sufficient data to perform accurate localization and/or classification.

In the current implementation and evaluation of the DELTA tracking group organization and maintenance performance, we have used light sensors to detect and track persons who are equipped with a flashlight. The target application scenario is terrain observation during night time. Persons moving around with flashlights shall be detected and tracked. Based on its movement pattern, an alarm must be triggered if the person is determined to be present without authorization. Later, we have further considered Passive Infrared (PIR) and vibration in the office monitoring application (see Chapter 9). In the current implementation, DELTA collects light information emitted by flashlights and processes the data at a dedicated leader node. Reports are sent to a management station in a fixed network, where the event reports can be stored or additional pattern recognition techniques can be applied to determine whether the person is present legitimately or not.

If movement patterns have to be identified online, the sensor network has to provide meaningful data in real time. Therefore, a fully distributed approach has been used for developing DELTA. This avoids heavy data load to the base station, which saves communication on the one hand and avoids congestion towards the base station on the other. The distributed implementation also leads to better communication load distributions, which additionally disburdens nodes close to the base station. The basic operations of DELTA are depicted in Figure 6.1.

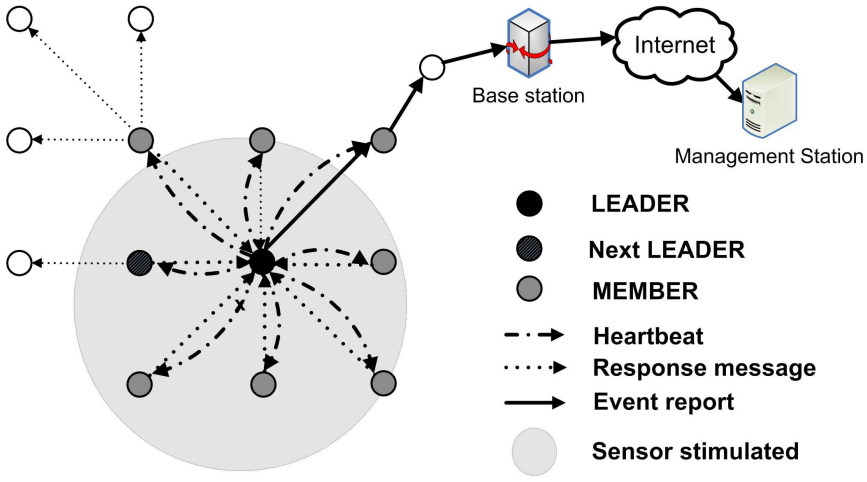



**Figure 6.1:** Event detection, tracking and reporting with DELTA.

A measurement-based leader election algorithm determines a unique group leader (the black node in Figure 6.1) which is responsible for group organization, group maintenance, event data processing and event reporting to the base station. The base station is connected to the Internet where the data is stored and/or further processed. The leader election algorithm implements a timer which is set accord-

ing to the amplitude of the signal(s) measured on the sensor node. The timer of the sensor node with highest sensor reading(s) expires first. The respective sensor node becomes leader and informs its neighbors about its state. The group member nodes (colored in gray in Figure 6.1) report their individual sensor readings. Thus, the leader is provided with the information required to perform localization and classification tasks. Moreover, all nodes which are located two hops away from the leader are thus informed about the existence of a leader and are prevented from establishing concurrent tracking groups. In Figure 6.1 this would apply to the two circled white nodes which are within sensing area, but not within transmission range of the leader.

DELTA has been designed to run on tiny sensor nodes. It has been implemented on the ESB sensor node platform (see Section 3.8.1). In the current implementation, DELTA is used to detect and track single events. There are no restrictions on the detection and tracking of multiple events as long as they occur in spatial sufficiently disjoint areas. If the event areas overlap, further statistical techniques might be necessary. Finally, DELTA requires that the sensor nodes know their own location. This can be achieved by GPS or other location services [73], [125]. Considering static networks with a predefined topology (e.g., monitoring of stockrooms), the node positions could even be set before or while deployment.

## 6.2 Distributed Group Formation and Maintenance

A key problem of event detection and tracking is the complexity of identifying and organizing the event relevant sensor nodes in a distributed manner with as little communication overhead as possible, while providing a satisfactory degree of accuracy. In many tracking applications the location of the event occurrence might not be predictable. Moreover, depending on the emitted event amplitude a large event area could result. Also, the event might move fast, possibly performing a sequence of successive shifts in direction. Such properties are difficult to predict and challenge any generic event detection and tracking algorithm.

In order to deal with generic and frequently changing conditions, DELTA considers the sensor measurements in the group setup and maintenance tasks. Thus, tracking groups can be efficiently managed. The common assumption that the communication range of the sensor nodes is significantly higher than the sensing range is overcome with DELTA: As soon as a leader evolves, it informs its local neighborhood about its state with a periodically sent notification message. This notification message is confirmed by the neighbor nodes by feedback messages that contain event-relevant sensor readings measured on the neighbor nodes. These sensor readings are required for the leader to perform event localization and classification. As a positive side-effect these feedback messages are overheard by all two-hop neighbors of the leader, which are thus implicitly informed about the existence of the leader. If required, the presence of the leader could be disseminated deeper into the network by rebroadcasting passive heartbeats (see Section 6.2.2).

### 6.2.1 Architecture Overview

To localize and track a moving event in a distributed manner some collaboration among the network nodes is needed. To achieve this, DELTA assigns different roles to the nodes. The states and state changes of the individual nodes and their roles are depicted in Figure 6.2.

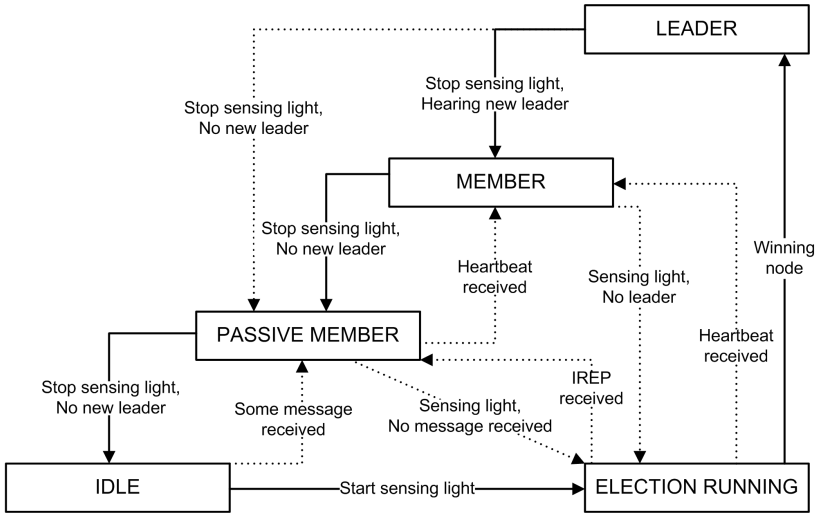



**Figure 6.2:** State diagram of the node roles of DELTA.

One sensor node is the leader of a tracking group. The leader is responsible for maintaining group coherence, performing localization and classification tasks, and communication with the base station. All direct neighbors of the leader are group members and deliver their relevant tracking and localization data to the leader. All other sensors are either passive members or idle. The passive member state has been introduced to inform the neighborhood of an event tracking group about a possibly upcoming event. Moreover, confusion caused by state switches can thus be prevented. A node enters leader election state if no communication is overheard but an event has been sensed. The node that wins the election process, i.e., its timer expires first, becomes leader and informs its neighbors immediately with a heartbeat message. In DELTA all roles are assigned dynamically. Independent from their state, all sensor nodes periodically check their sensors to provide an appearing leader with their information upon request.

The solid black lines in 6.2 show the normal sequence of state changes of a leader node when an event occurs in its sensing area and later moves away. When sensing the event, the subsequent leader enters leader election state. Winning the contention, the node becomes leader and manages the group until the event leaves its sensing area. The leader optionally initiates a leader handover and becomes a member of the subsequent group. In Figure 6.1 the leader state will be handed over to the member node on the left side of the current leader. The current leader node will likely become a member of this subsequent tracking group, because it will still

be in the event sensing area and it is adjacent to the new leader. As the event moves farther away, the old leader switches from member state to passive member state and finally to idle mode.

### 6.2.2 Leader Election and Maintenance

All network nodes remain in idle state unless an event is sensed. As soon as an event has been observed, every sensing node switches to election running state and schedules a timer according to the sensed signal amplitude(s). The timer expires the sooner the stronger the event has been sensed at the respective node. When the timer expires, a heartbeat message is broadcast to inform the neighborhood about the presence of a group leader. All receiving nodes immediately cancel their own timer and become group members. An appropriate setting of the timer is crucial because it determines the leader node. The current timer setup partly depends on the applied hardware and is therefore described in detail in Section 6.3.

The leader node initializes and maintains several variables which are required to manage a group. The observed event is identified by a temporary unique event tag. It is used to announce the tracking group to the base station as well as to maintain group coherence. To avoid processing of outdated information a round number is introduced. The round number is incremented on the leader whenever a heartbeat message is sent. The current round number is included in every control message. Every message with a round number smaller than or equal to the current round number is discarded. A TTL field defines the depth, until which the leader state is disseminated into the network, i.e., it determines the size of a tracking group in hops. The leader node is furthermore responsible to support a controlled leadership handover if an event leaves the sensing area of the leader. If a handover is required, the leader node immediately broadcasts a leader reelection message. Optionally, a leader node can determine its successive leader node, e.g., by computing the event location and electing the sensor node closest to this location as successor. The successive leader is also communicated within the leader reelection message.

Considering events where the sensing range is larger than the communication range, not every node that senses the event is a direct neighbor of the leader (e.g., nodes C and D in Figure 6.3). Accordingly, these nodes are not informed by the heartbeat messages. However, the information response (IREP) messages, which are required to report the location and classification relevant data of the group members to the leader, inform all nodes two hops away from the leader node. In most cases this is sufficient to cover the sensing area completely. If even larger sensing ranges are present, a passive heartbeat mechanism might be used to inform nodes farther away about the existence of an event (e.g., node D in Figure 6.3). Of course, this implies some overhead. Optimized broadcasting techniques might be used [50], [155]. With the proposed mechanism the message flow of DELTA overcomes the restriction $\frac{SR}{CR} < 1$ or even $\frac{SR}{CR} < \frac{1}{2}$ as illustrated in Figure 6.3.

Multiple, concurrently present tracking groups lead to confusion in the network and communication overhead. In particular into the direction of the base station

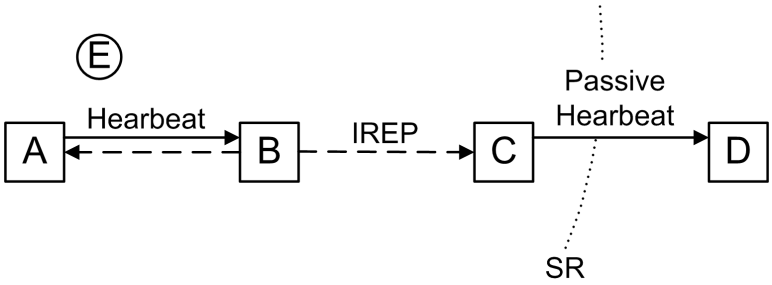



**Figure 6.3:** Group communication in a DELTA tracking group.

network nodes are unnecessarily strained by having to forward redundant event reports. Concurrent leaders are well prevented by the communication mechanism implemented in DELTA, though.

The leader election process aims at quickly determining a single leader node which is able to cover a moving event reliably and as long as possible. Frequent leader reelections and leader handover lead to confusion and periods where no leader is available. To minimize the number of handover, the leader located closest to the event position is elected. This is the node with highest event sensing amplitudes (highest sensor readings). Because the direction of the event is basically not known, the node closest to the event is the best choice to optimize monitoring time. The elected leader node should further have enough remaining battery power to bear the burden of temporary increased communication and computation load. Finally, the election process needs to be fast to avoid periods when no leader is present. DELTA has been developed to satisfy these requirements by implementing the leader election timer based on measured sensor readings.

## 6.3 Implementation Details

The ESB sensor nodes have been used (see Section 3.8.1) for the real-world evaluation. Moving persons are detected and tracked according to the light they emit with flashlights. To sense the emitted signals more accurately, an exponentially weighted moving average filter has been implemented: $\overline{x}_k = \alpha\overline{x}_{k-1} + (1-\alpha)x_k$. The computation of the average $\overline{x}_k$ requires only the storage of the past value $\overline{x}_{k-1}$ and the actual light measurement $x_k$. Any change in the measured signal that varies more than a configurable threshold T from the average is considered as significant and the signal is processed by the sensor. Currently, T is set to 50. In addition to noise filtering the moving average filter supports a slow adaptivity to changing brightness in the environment. The moving average filter converges to the current brightness. Thus, permanent throwing of events in bright environments (e.g., during the day or during building works) is prevented. An $\alpha$ value of 0.9 has been chosen in the current implementation.

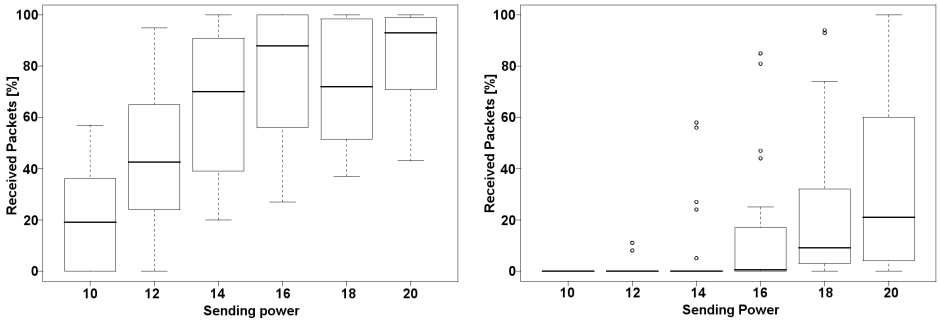
As mentioned in Section 6.2.2, the computation of the leader election timer is

crucial for the performance of DELTA. On the ESB platform we calculate the light irradiance every 200 ms for exactly 100 ms. Because the TSL245 output frequency is limited to 100kHz (see Section 3.8.1), the light values range from 0 to 10'000. The timer should be set according to the level of irradiance and is computed as follows:

$$\begin{aligned} \Delta t[ms] &= \frac{I_{MAX} - I_C}{10} \\ \Delta round[ms] &= round(i) \cdot SAMPLE\_FREQUENCY \\ \Delta t &= \begin{cases} \Delta t & , \Delta t < \Delta round \\ \Delta t = \Delta t - \Delta round & , \text{else} \end{cases} \end{aligned}$$

where $I_C$ is the currently measured irradiance. $I_{MAX}$ is the maximum allowed irradiance of 10'000. Accordingly, $\Delta t$ computes a delay between 0 and 1 second. SAMPLE_FREQUENCY is the sensing frequency of 200ms. The round variable is set to 0 when the election is initialized and incremented in each subsequent sensing step. The proposed computation of $\Delta t$ supports the filtering of non-continuous irradiance peaks as long as the value is not too high, i.e., if the timer does not expire before the next light sensing is performed.

The ESB sensor nodes support transmission power control. Transmission power control can be used to save energy while restricting communication to single-hop communication in specific topologies. In our experiments, neighbor nodes are placed 1.25 meters away from each other. Therefore, the transmission power has been adjusted to cover a range of approximately 1.75 meters. Transmission power control results are shown in Figure 6.4.




**(a)** Distance 1.25 meters

**(b)** Distance 2.5 meters

**Figure 6.4:** Fraction of received messages for varying transmission power.

Two distances of 1.25 and 2.5 meters between a sending node and some receiving nodes were tested. At each receiver location four receiver nodes were placed. The transmitting node was replaced four times with different sensor nodes. Each of

the four transmitting nodes transmitted 100 packets of a size of 35 bytes. The transmission power was increased from 10 to 20. The maximum transmission power of the ESB is 99. The overall fraction of received messages is shown in Figure 6.4. From this evaluation we conclude that a transmission power of 16 is the best choice for the current network topology. A high fraction of packets was received at a distance of 1.25 meters, while only few packets arrived at a distance of 2.5 meters. Setting the sending power to a lower level would involve too much packet loss at 1.25 meters. On the other hand, higher levels would result in too high receive fractions at a distance of 2.5 meters.

In dense networks the burst of IREP messages is handled inefficiently by CSMA with random backoff. Limiting factors are the required 2 ms to switch an ESB node from receive to transmit state and the approximately 14 ms needed to transmit an IREP message at 76kbps. On the other hand, the leader requires only a certain number of IREP messages to estimate the event position. Therefore, we have implemented a simple, on-demand time division multiple access (TDMA) mechanism on the nodes: Within the heartbeat message the leader schedules at most $n$, with $n \leq 8$, members. The leader learns these members from IREP messages it has received in previous rounds. In any subsequent communication any member node addressed in the IREP message responds in the time slot it has been assigned to by the leader. Thus, all addressed nodes can send their data in a collision-free way in the first $n \cdot 14$ ms. As long as not all slots are occupied, all non-addressed member nodes use common CSMA with random backoff after the TDMA period to transmit their data. If all slots are occupied, the sending of messages by non-addressed nodes after the TDMA period can optionally be switched off. Of course, this optimization works only if the event is sufficiently long, i.e., a few number of times, within sensing area of the currently responsible leader node.

## 6.4 Evaluation

The DELTA algorithm has been evaluated both in simulations and in real-world experiments. Simulations allow a faster development of the platform-independent functionality and simplify the evaluation of larger networks. DELTA has been compared to EnviroTrack (see Section 3.4.5). EnviroTrack provides similar group maintenance and tracking functionality. EnviroTrack groups do not collect sensor readings and are thus not able to support localization and classification.

### 6.4.1 Simulations

The simulation settings have been adopted from the original EnviroTrack simulation. The goal in [1] was to track T-72 battle tanks moving through an off-road environment. For the simulations a realistic object path, neither with sharp turns nor following just a straight line, was used (see Figure 6.7). DELTA has been evaluated with a TTL of 1 (only heartbeats are sent like in EnviroTrack) and a TTL of 2 (reporting event relevant data and informing the two-hop neighborhood about a

leader). The speed of the target object and the ratio between sensing range (SR) and communication range (CR) were varied. Any experiment was repeated eight times. The mean, minimum and maximum values are shown in the results. The sensor network consisted of 160 nodes arranged in a grid consisting of 8 x 20 nodes. The distance between any two neighbor nodes was set to 25 meters. The prevention of concurrently present leaders is of main interest.

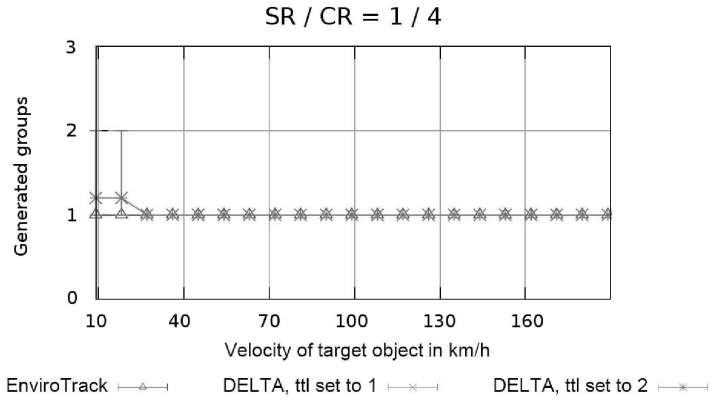



**Figure 6.5:** Tracking of small-area events.

Figure 6.5 shows results with the communication range being significantly higher than the sensing range. Such scenarios are tailored to EnviroTrack and all protocols perform equally well. Considering the ratio of $\frac{1}{4}$ between sensing range and communication range, it is not surprising that DELTA with the TTL set to 1 performs equally well as when the TTL is set to 2. In such scenarios, groups can easily be organized by the heartbeat mechanism alone. Of course, neither localization nor classification can be performed if only heartbeat messages are used.

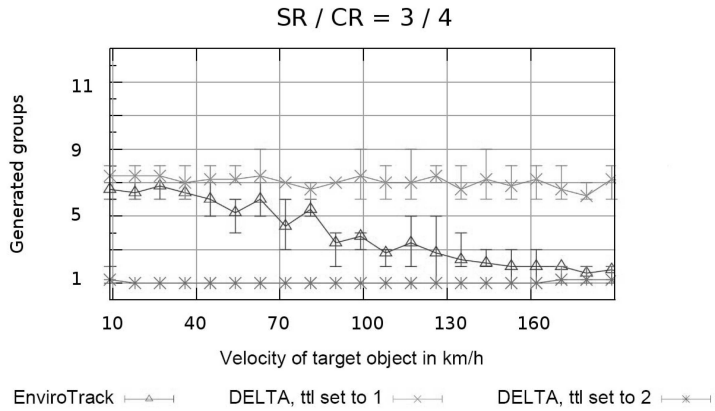
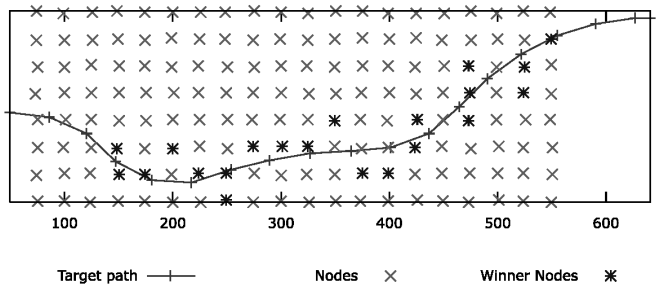


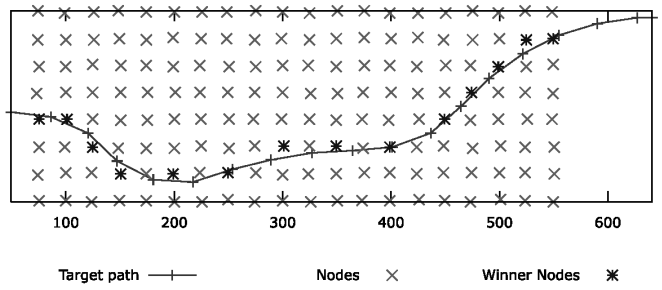

**Figure 6.6:** Tracking of large-area events.

Figure 6.6 shows the performance results if the ratio between sensing range and

communication range is larger than $\frac{1}{2}$. Even with a ratio of $\frac{3}{4}$, which only slightly hurts the constraint of EnviroTrack that the sensing range needs to be smaller than half the communication range, the number of coexisting groups increases considerably for both EnviroTrack and DELTA with the TTL set to 1. This shows that in scenarios with high sensing ranges a passive heartbeat mechanism alone is not sufficient. Answering heartbeat messages with IREP messages solves the problem of concurrent leaders and supplies the leader with the information needed to support localization and classification. The decreasing number of leaders in EnviroTrack for higher speeds is due to the inability of EnviroTrack to build groups in time.




**(a)** EnviroTrack.




**(b)** DELTA.

**Figure 6.7:** Nodes elected as leader in a specific simulation run.

Figure 6.7 shows two typical runs of DELTA and EnviroTrack if the ratio between sensing range and communication range is $\frac{1}{4}$. The moving object crosses the network at a high velocity of 135 km/h. The transmission range is 100 meters. Accordingly, up to four neighbor nodes in one direction can be informed by the heartbeat message. The figures show the effectiveness of the leader election timer. The random timer set by EnviroTrack is not able to determine a leader node (winner node) in time in the beginning, as shown on the left in Figure 6.7(a). Moreover, the election of leader nodes is, in respect to the path of the moving object, less optimal with EnviroTrack than with DELTA. Finally, the leader election based on the

sensor readings optimizes the number of required leaders. In the next section the performance of DELTA and EnviroTrack in a real sensor network is investigated.

### 6.4.2 Real-World Experiments

All tests were performed indoor in a darkened room to minimize external influences. 25 nodes were arranged in a grid consisting of 5 x 5 nodes, with a spacing of 1.25 meters between every pair of nodes. The setup is depicted in Figure 6.8.

**Figure 6.8:** Experiment setup with 25 sensor nodes.

The transmission power was reduced to a value of 16 to restrict communication only to neighboring nodes. Two lamps, common office equipment with a 25 W bulb and a 40 W bulb, were used as light sources. The lamp was held approximately 1.5 m above ground pointing to floor 1.5 m in front of the moving person. Thus, the directly illuminated area was a circle with a diameter of approximately 2 m for the 25 W bulb and 4 m for the 40 W bulb, respectively. The person covered a distance of 7 m, walking at a constant speed of about 0.3 m/s.

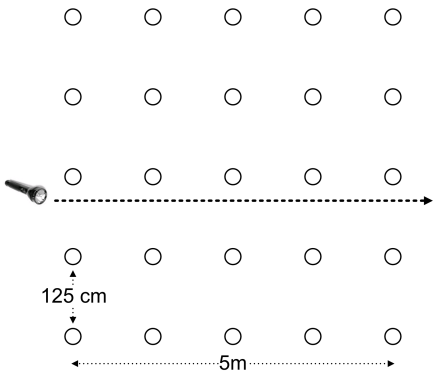


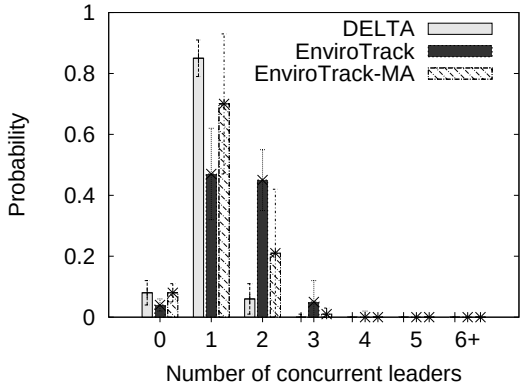

**Figure 6.9:** Event path through the sensor network.

The person walked along a straight line through the sensor network as illustrated in Figure 6.9. Each experiment was repeated five times and a 95% confidence

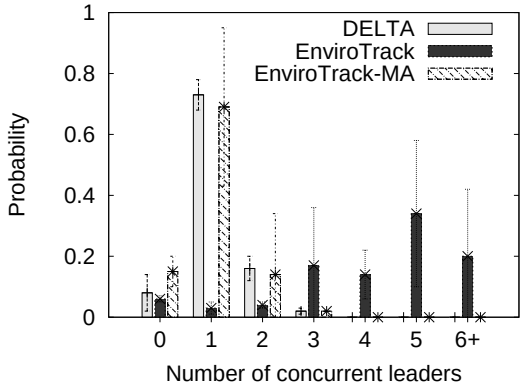
interval was applied. To be able to better assess the impact of the different enhancements of DELTA, a second EnviroTrack version (EnviroTrack-MA) enhanced with a moving average filter (see Section 6.3) was implemented too.

### Number of Concurrent Leaders

The results of the detection and tracking performance of DELTA and EnviroTrack are shown in Figure 6.10.




**(a)** Tracking of a 25W bulb




**(b)** Tracking of a 40W bulb

**Figure 6.10:** Fraction of concurrent leaders in DELTA and EnviroTrack.

If the sensing range increases (40W bulb), DELTA produces significantly fewer concurrent leaders than the original EnviroTrack implementation. This supports the simulation results. Concurrent leaders produce unnecessary event reports, producing confusion while wasting energy and bandwidth. The network load towards the

base station is increased, affecting the overall network lifetime.

The performance of EnviroTrack enhanced with the moving average (MA) filter is nearly as good as the performance of DELTA. The fast convergence of the MA filter at nodes close to the border of the sensing area, i.e., at nodes that measure only small signal magnitudes due to their location, suppresses them from being elected as leaders. Nevertheless, EnviroTrack is not able to support any accurate localization or classification. Moreover, there is a slightly higher fraction of time when no leader is present. This is due to the random leader election timer that more often elects poorly located nodes as leader than the measurement-based timer implemented at DELTA nodes.

### Communication Costs

In the current implementation DELTA supports the localization and classification of events in the plane. The according optimization problem has a dimensionality of 3 due to unknown position and signal amplitude (see Section 3.5.1). To solve the according optimization problem sensor readings from at minimum three group members are required (IREP messages). If a nonlinear optimization method such as Simplex Downhill is used the minimum amount of information is sufficient to get useful estimations (see Chapter 7). For this reason and in order to minimize communication costs, the transmission of IREP messages was restricted to 3. Therefore, the on-demand TDMA mechanism proposed in Section 6.3 was used. In theory, setting the number of available slots for IREP senders to 3 should be sufficient. However, due to packet loss, on average 5 responding members were required to guarantee the reception of 3 IREP messages using ESB nodes.

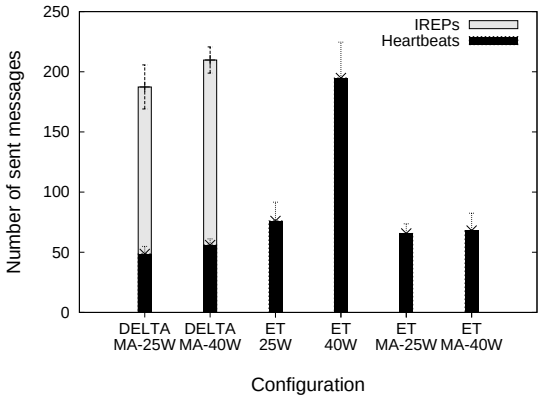



**Figure 6.11:** Communication costs of DELTA and EnviroTrack.

Communication costs of DELTA and EnviroTrack are depicted in Figure 6.11. Figure. 6.11 shows that for a higher sensing range the communication costs of DELTA are similar to those of EnviroTrack. However, DELTA already provides the leader with the information needed for localization and classification. On the

other hand, similar functionality would have to be added to EnviroTrack to support localization and classification.

EnviroTrack enhanced with the MA-filter was able to keep the number or sent messages small. Accordingly, communication costs for group maintenance were decreased. No localization and classification can be supported yet, though. In general, DELTA elects leaders located closer to the event than EnviroTrack. These nodes cover the sensing area well. On the other hand, with EnviroTrack MA slightly more concurrent tracking groups occur due to the election of poorly located leader nodes. This leads to the slightly higher amount of sent heartbeats observed with EnviroTrack MA. Therefore, if neither localization nor classification is required, DELTA with the MA filter but without IREP functionality is the best choice.

The DELTA results show that on average the desired number of 3 IREP messages has been received. Accordingly, localization and classification methods can be executed on the leader. Of course, parameters can be adapted to collect more information about the present event.

## 6.5 Conclusions

The DELTA algorithm provides an efficient and fast event detection and tracking algorithm. Tracking groups are efficiently and dynamically created. DELTA works in many cases including smart dust environments with small radio ranges and high sensing ranges. The leader election procedure of DELTA is adaptive, quick and precise. Using the sensor readings improves both, event detection and tracking performances. The implementation of a moving average filter allows the suppression of poorly located sensor nodes. The convergence property of the filter is dependent on the platform and network used and therefore needs to be considered before deployment.

DELTA provides the event detection system with the required detection and tracking functionality. Additionally, DELTA provides the overlaid localization and classification mechanisms with the necessary information. It has been shown that the usage of 5 IREP transmission slots together with the MA filter was sufficient to collect the required data and to suppress concurrent leaders (see Figure 6.10). In later localization experiments the TmoteSky nodes (see Section 3.8.2) were used, too. With TmoteSky nodes more than 95% of all IREP messages were received. Accordingly, on TmoteSky nodes the number of slots for IREP senders (members) could indeed be decreased to the minimum of 3 to receive 3 IREPs in average. To conclude, DELTA was shown to optimize the desired trade-off between required accuracy and communication cost minimization. In the next chapter the localization of events based on the information collected by DELTA is presented.

Chapter 7

# Event Localization and Signal Strength Estimation

This chapter discusses the localization and signal strength estimation features provided by DELTA [150], [152]. In the previous section it has been shown that events are observed and tracked by dynamically established groups. Relevant sensor data is collected at appointed nodes (group leaders) which are destined to perform all subsequent localization and group organization tasks. Both, event position and the signal strength(s) of the emitted signal(s) of an event are estimated. The emitted signals and their amplitudes are event-characteristic and can be used for classification of the respective event sources.

## 7.1 Introduction

In this chapter DELTA is enhanced with the localization and signal strength estimation logic, which is based on a well-known sensor model. The model has been introduced in Section 3.5.1. The solution of the localization and signal strength estimation problems is addressed in Section 7.2. Section 7.3 provides simulation and real-world performance results.

Existing localization and signal strength estimation approaches mainly focus on accurate but cost-intensive collaborative signal processing (CSP) methods. The related work has been presented in Section 3.5. Energy-efficient performance and network organization issues as proposed in our own approach, presented in the previous chapter, are barely considered. On the other hand, depending on the application, localization and classification requirements might be less constrained than in traditional CSP research fields such as robotics. Therefore, in wireless sensor networks more cost-efficient methods might be sufficient. DELTA bridges the gap by providing satisfying accuracy while keeping the network load at a reasonable level. The performance of the proposed localization and signal strength estimation methods is evaluated in simulations as well as on real hardware. Moreover, a problem of closed-form linear least-square solutions is outlined and discussed.

Based on the information collected at the leader node, DELTA is able to esti-

mate both, the location and the characteristics of an event, i.e., the emitted signal power(s) of the event. There are two restrictions on the kind of signals which can be used in the localization and classification procedures. First, the computation of position and signal emission power requires an attenuation model for each considered signal (e.g., sound, vibration, RSSI). Second, to be able to classify distinct events, the signal emission power of every specific event needs to be characteristic, i.e., more or less constant. Considering classification, which is addressed in Chapter 8, an accurate determination of the event position is of little importance. There, the event location is mainly derived as byproduct in the emitted signal power computations. For other applications the event location might be of different interest.

## 7.2 Event-Based Localization and Signal Strength Estimation

Event-based tracking and localization methods use the sensor readings collected on the sensor nodes. In order to assess the collected information, adequate sensor models for the particular sensors are required. The commonly used sensor model has been presented in Section 3.5.1. In our work, this model has been used too. The resulting nonlinear optimization problem can either be solved with nonlinear methods (see Sections 3.5.2 (Simplex Downhill) and 3.5.3 (Conjugate Gradient method)) or it can be linearized and solved (see Section 3.5.4).

In the simulation part of this work we will show that the linearization lacks drastic accuracy if the linear system is not over-determined. On the other hand, over-determination requires additional communication, which cannot always be provided in wireless sensor networks. Moreover, even in over-determined systems linear least square methods are not able to provide the similar degree of accuracy as nonlinear methods.

Accordingly, for DELTA we evaluated the Simplex Downhill and the Conjugate Gradient descent method. Both algorithms are not protected against finding local minima. Therefore, the determination of a well-placed starting point, respectively simplex, is crucial. Finding the global minimum is a challenging problem. Moreover, it is very cost-intensive and therefore not suitable for our purposes, i.e., it needs an additional search procedure (e.g., Monte Carlo) which makes it unfeasible to run on sensor nodes.

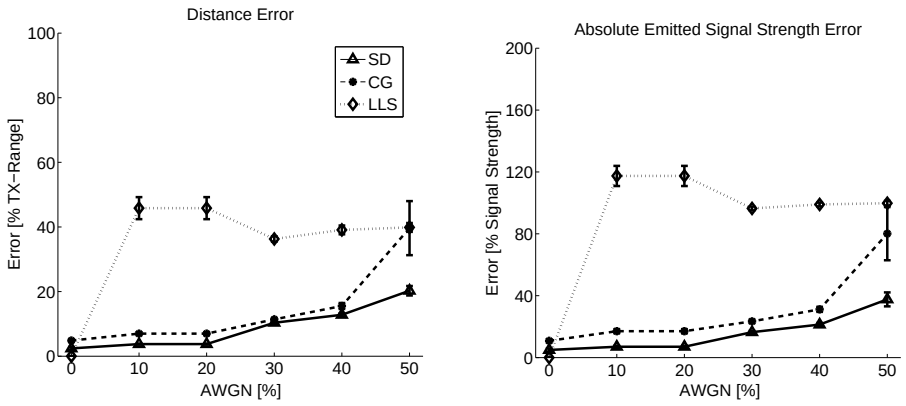
## 7.3 Evaluation

DELTA provides the leader node with the information needed to localize and classify events. In a first step, different possible localization methods have been evaluated in Matlab. The Simplex Downhill (SD) and the Conjugate Gradients (CG) methods together with a closed-form linearized least square (LLS) solution have been considered. The Simplex Downhill method showed the best performance and

was chosen for subsequent implementation on real sensor hardware. Results are presented in the subsequent sections.

### 7.3.1 Simulations

In a first evaluation four nodes were arranged in a square with a side length of 125 cm. Thus, all localization methods had to work with the minimum amount of data that is required to solve a system with three unknowns (x, y, c). The linear least square method was expected to perform poorer than the nonlinear methods, because the system is not over-determined. An event was randomly placed within this square. The localization was performed 200 times with a confidence interval of 95%. The intervals are very small and thus difficult to observe in the figures. Both, Simplex Downhill and Conjugate Gradient require well located starting points. For Simplex Downhill the simplex is located at the center of area of the sensing nodes and their measurements. For Conjugate Gradient only the center of area is needed. Optionally, the linear least square method could be applied to determine the starting point, respectively simplex, of the Simplex Downhill and Conjugate Gradients methods. Noise in the sensor readings is modeled as Additional White Gaussian Noise (AWGN). AWGN distorts the received signal according to normal distributions. The AWGN level has been increased in steps of 10% from 0 to 50%.




**(a)** Distance Error **(b)** Signal Strength Error

**Figure 7.1:** Localization and Signal Strength Estimation of SD, CG and LLS.

The localization and signal strength estimation results are shown in Figure 7.1. The results show that the linear least square method does neither compute the location of the event nor its emitted signal amplitude satisfactorily. Almost independent from the noise level, the position error is always about 40% of the transmission range (i.e., of the grid length). The signal amplitude error is even worse. On the other hand, both Simplex Downhill and Conjugate Gradient perform well even if

only the minimum amount of information is available as in Figure 7.1. Due to unreliability and efficiency reasons this might happen frequently in sensor networks.

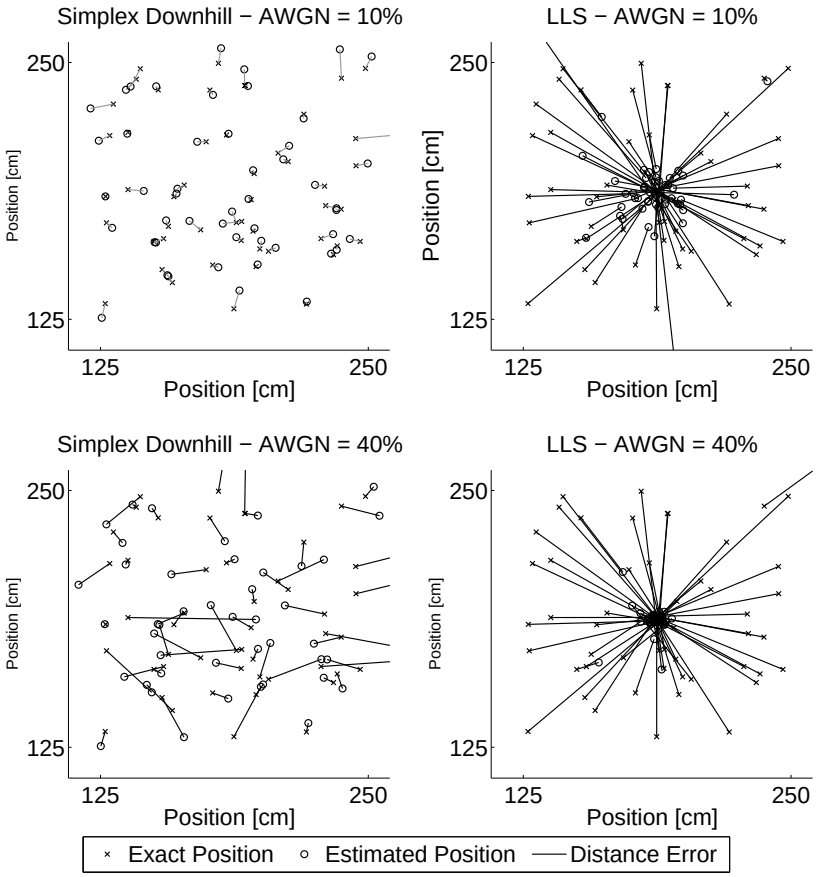
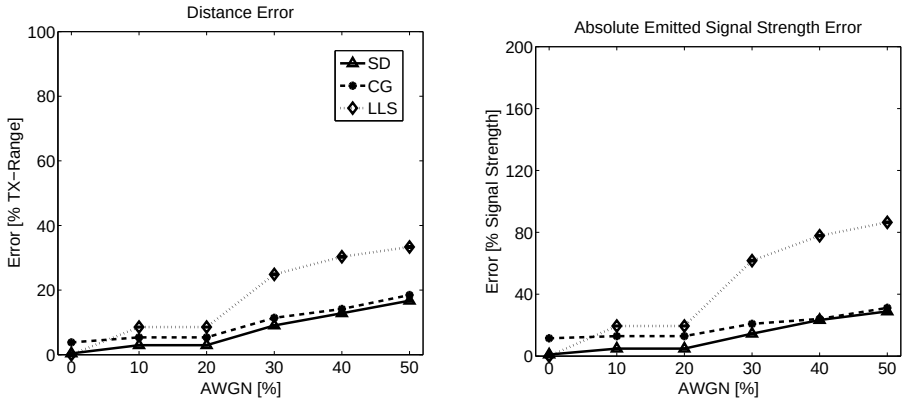



**Figure 7.2:** Localization accuracy of SD and LLS.

The distance errors of Simplex Downhill and linear least square are shown in Figure 7.2 for AWGN of 10% (good signal to noise ratio) and 40% (poor signal to noise ratio), respectively. To improve readability, the distance errors of only 50 out of 200 position estimations are depicted. Only little affected by AWGN, the majority of the linear least square estimations are close to the center of the sensing area. The distance errors (lines between the exact event positions and their estimations in Figure 7.2) are accordingly high. Considering the Simplex Downhill method, the noise level has a noticeable impact. Nonetheless, even with a noise level of 40% the accuracy of the estimated event location might be sufficient for most applications. The accuracy of the linear least square method can be improved if the system is over-determined, i.e., if more than four sensor nodes are used in the scenario above. Of course, this implies additional communication load. Moreover,

due to increased collision and link failure probabilities it might become challenging to collect that information (see also Section 6.4.2).




**(a)** Distance Error **(b)** Signal Strength Error

**Figure 7.3:** Accuracy of LLS, SD, and CG in an over-determined system.

In Figure 7.3 results obtained with 6 sensor nodes (over-determined system) are shown. The two additional nodes were placed at positions (175, 125) and (175, 250). As expected, the performance of linear least square is improved in the over-determined system. However, it does still not achieve the performance of the non-linear methods. The distance errors of Simplex Downhill and linear least square are shown in Figure 7.4 for noise levels of 10% and 40%, respectively. Again only 50 out of 200 position estimations are depicted in order to improve readability.

In conclusion, both Simplex Downhill and Conjugate Gradient outperformed the linear least square method in all scenarios. Moreover, with a nonlinear solution it is possible to solve localization and signal strength estimation problems with the minimum amount of information required, which implies less communication load and a higher success probability. The performance of Simplex Downhill shows a good trade-off between estimation accuracy and communication load to provide the leader node with the required information.

### 7.3.2 Real-World Experiments

Based on its good performance in the simulations and its simplicity, the Simplex Downhill algorithm was chosen for experiments on real hardware. Real-world experiments were performed on two different platforms, namely on ESB sensor nodes (see Section 3.8.1) and on TmoteSky sensor nodes (see Section 3.8.2). In a first set of experiments the Simplex Downhill method was implemented as an add-on to DELTA on the ESB sensor nodes. In subsequent experiments we focused only on localization and signal strength estimation performance and implemented the ac-

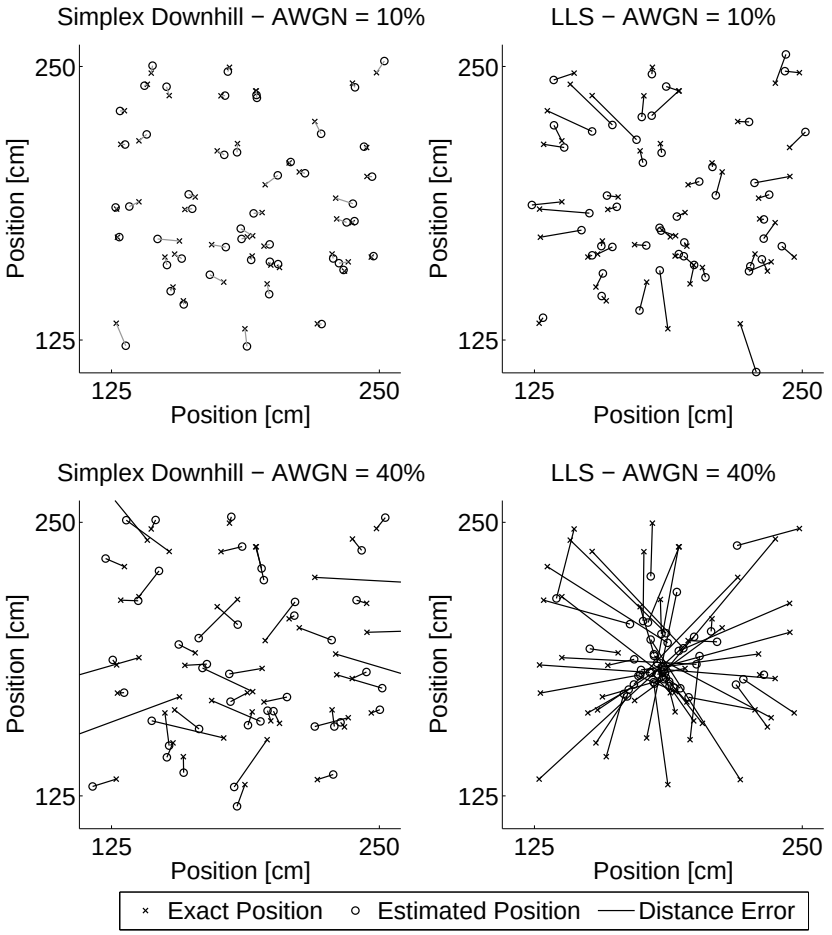



**Figure 7.4:** Localization accuracy of SD and LLS in an over-determined system.

cording functionality on the TmoteSky nodes. The results of the latter experiments were used for subsequent classifications.

### Implementation on ESB Nodes

For the real-world experiments on the ESB nodes the same network setup as for the simulations was used. The Simplex Downhill algorithm was implemented according to [114]. Similar to the experiments performed in the detection and tracking evaluation of DELTA in Section 6.4.2, two light sources of 25 Watt and 40 Watt were used. Every estimation was repeated for 50 times. The localization frequency was two times per second. In contrast to the simulations, the event was not randomly placed in the event area, but at specific positions: P1(250, 250), P2(250,188), P3(188,188), and P4(219,219). The sensor node locations (**o**) and the event locations (**x**) are shown in Figure 7.5.

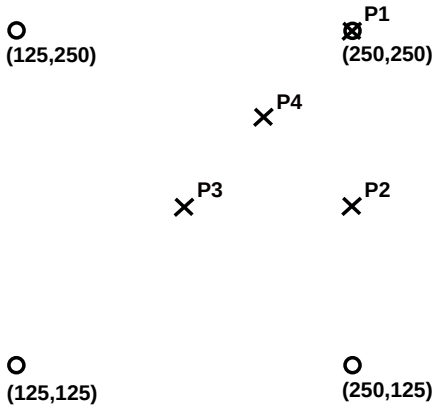



**Figure 7.5:** Arrangement of sensors **o** and event locations **x**.

The mean distance errors ($\varnothing$) and the standard deviations ($\sigma$) of the localization tests are provided in Table 7.1. Considering the distance of 125 cm between two neighbor nodes, a maximum mean location estimation error of 21 cm, at location P1 using the 40 Watt bulb, is acceptable. The Simplex Downhill method performs best for locations inside the square. The performance is decreased if the event position is very close to a sensor node. The standard deviation was small in all experiments. In a large sensor network the sensor nodes observing an event will in general be located around the event source. Thus, the event positions should in most cases lie within coverage area of the monitoring nodes.

**Table 7.1:** Distance errors $\varnothing$ and standard deviations $\sigma$ on ESB nodes in [cm].

| | 25 Watt | | 40 Watt | |
|---|---|---|---|---|
| Position | $\varnothing$ | $\sigma$ | $\varnothing$ | $\sigma$ |
| P1 | 18.43 | 0.14 | 20.91 | 0.23 |
| P2 | 3.86 | 0.59 | 14.94 | 3.21 |
| P3 | 6.3 | 0.85 | 4.13 | 0.11 |
| P4 | 3.69 | 1.6 | 5.04 | 1.68 |

Apart from the position, the Simplex Downhill method also computes the emitted signal strength of the event source. For the classification of events this value is even more important than the event position, as it is assumed to be characteristic for the event. Due to the implementation of the light sensor on the ESB sensor nodes, light intensity is estimated in Hz (see Section 3.8.1). The mean emitted signal strength computed for the 25 Watt bulb is $1.71 \cdot 10^6$ Hz with a standard deviation of $0.246 \cdot 10^6$ Hz. On the other hand, the mean amplitude of the 40 Watt bulb is $2.88 \cdot 10^6$ Hz with a standard deviation of $0.452 \cdot 10^6$ Hz. Obviously, the resulting spectra of both events are disjoint and can therefore be used for classification.

### Implementation on TmoteSky Nodes

Compared to the ESB platform, the TmoteSky sensor nodes provide more reliable communication due to the implemented radio (see Section 3.8.2). Moreover, they provide Photosynthetic Active Radiation (PAR) and Total Solar Radiation (TSR) light sensors. In subsequent experiments the TSR and the PAR sensors were used to classify different light sources. For the localization, signal strength estimation and classification experiments five different light sources, bulbs of 25, 40, 60, 75, and 100 Watt, were used. The light bulbs were arranged according to the setup depicted in Figure 7.6.

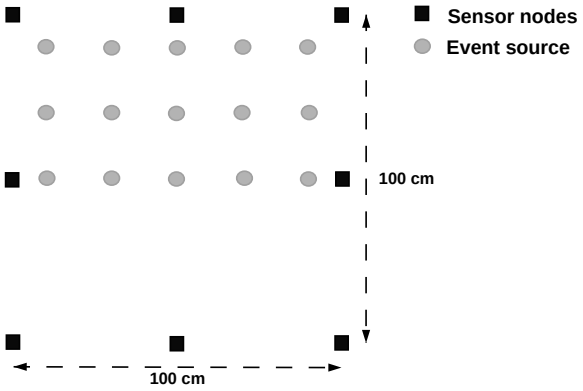



**Figure 7.6:** Experiment setup with 15 event positions.

In the experiments the light sources were placed at 15 locations. Each experiment was repeated 30 times, which resulted in sample set sizes of 2250 samples. According to DELTA (see Chapter 6) one of the sensor nodes in Figure 7.6 acted as leader node and requested the sensor readings from its neighbor nodes twice a second. The information collected by the tracking group depicted in 7.6 provided the leader node with sufficient information to compute event position and characteristics in an over-determined system. In the ESB experiments the sensor readings of 4 sensor nodes were considered. Accordingly, the ESB results were computed with the minimum amount of sensor readings. On the other hand, in the TmoteSky experiments 8 sensor nodes were considered. Therefore, performance on the TmoteSky nodes should be better. However, a comparison is difficult, because the sensor nodes implement different light sensors. The ESB sensor node implements a light sensor that measures only infrared (see Section 3.8.1). On the other hand, the PAR and TSR sensors implemented on the TmoteSky both include visible light.

Again, the Simplex Downhill algorithm was applied. The position of the event as well as the emitted signal strengths of the PAR and TSR values were computed. The estimates are reported to a base station, where the collected data is used for classification (see Chapter 8).

In Table 7.2 the mean distance errors $\varnothing$ and the standard deviations $\sigma$ of the

**Table 7.2:** Distance errors $\varnothing$ and standard deviations $\sigma$ on TmoteSky nodes in [cm].

| | PAR | | TSR | |
|---|---|---|---|---|
| Bulb | $\varnothing_1$ | $\sigma_1$ | $\varnothing_2$ | $\sigma_2$ |
| 25 W | 5.31 | 3.29 | 9.67 | 5.30 |
| 40 W | 4.57 | 3.54 | 13.67 | 7.10 |
| 60 W | 4.43 | 3.07 | 18.26 | 8.66 |
| 75 W | 5.03 | 3.35 | 20.94 | 9.54 |
| 100 W | 6.48 | 2.63 | 25.00 | 10.31 |

event position estimations are shown. In all experiments the distance estimation was more reliable with the PAR light sensor. Moreover, it remained more or less constant at approximately 5 cm for all tested light bulbs, which is a feasible accuracy considering the monitored area of 1 square meter. On the other hand, the estimations based on the TSR light sensor showed a larger variation over the different light bulbs. The distance errors varied from 9 to 25 cm. Accordingly, the PAR sensor is better suited for event localization purpose. However, if accuracy requirements allow it, the TSR sensor can be used too.

**Table 7.3:** Signal strength errors $\Theta$ and standard deviations $\sigma$ on TmoteSky nodes in Lux.

| | PAR | | TSR | |
|---|---|---|---|---|
| Bulb | $\Theta_1$ | $\sigma_1$ | $\Theta_2$ | $\sigma_2$ |
| 25 W | $3.24e^4$ | $0.77e^4$ | $3.16e^4$ | $0.47e^4$ |
| 40 W | $5.72e^4$ | $0.96e^4$ | $5.23e^4$ | $0.73e^4$ |
| 60 W | $9.61e^4$ | $1.18e^4$ | $7.94e^4$ | $1.17e^4$ |
| 75 W | $12.07e^4$ | $1.29e^4$ | $9.54e^4$ | $1.38e^4$ |
| 100 W | $17.47e^4$ | $1.89e^4$ | $12.09e^4$ | $1.64e^4$ |

For classification only the emitted light signal strengths are used. Table 7.3 shows the mean emitted signal strengths ($\Theta$) and standard deviations ($\sigma$) of the experiments. Considering the PAR sensor, it seems as if the different light bulbs should be distinguishable, at least to some extent. This means the mean values together with the respective standard deviation are more or less disjoint. The same is true for the TSR. However, similar to the distance estimation errors, the variations are again higher for the TSR sensor.

Comparing both sensor platforms in terms of localization and signal strength estimation accuracy was not in the scope of our work. The ESB sensor nodes have been used to develop DELTA and to provide localization and signal strength estimation results. On the other hand, the TmoteSky platform has then been used to evaluate the classifiers. Nevertheless, the PAR sensor provides slightly better results than the infrared sensor implemented on the ESB sensor nodes. This might

partly be due to the different hardware, but is partly also due to the usage of additional sensor readings in the TmoteSky experiments. The TSR sensor has performed similar to the infrared sensor implemented on the ESB sensor nodes.

## 7.4 Conclusions

The DELTA localization and signal strength estimation functionality provides accurate and distributed optimization methods. The evaluations showed that a nonlinear algorithm is best suited in terms of communication load and accuracy. Based on the Simplex Downhill optimization method, estimates of the position and of the emitted signal strengths of an event can be computed on a leader node. The accuracy of the mechanisms is satisfying. In particular with the PAR sensor implemented on the TmoteSky nodes, distance errors were in the order of 5% of the transmission range, which is sufficiently accurate for many applications. Furthermore, the computed emitted signal strength(s) of the different light sources are more or less disjoint. Thus, classifications based on these estimations are possible. The computed estimates can be transmitted in event reports to the base station. Thus, DELTA avoids communication overhead in the event observing area and even more important on the paths towards the base station. The classification of events is addressed in the next Chapter.

Chapter 8

# Event Classification and Reasoning

This chapter considers event classification and reasoning. Event reports, collected and computed in a distributed manner with DELTA, are used to monitor and classify observed events [152], [153], [154]. In [152] the classification of time-discrete events is considered. The respective training and test data were collected with DELTA. In monitoring tasks such as anomaly detection in office access, events are not present as discrete entities, but occur over time. Hence, methods to model and process such events are required. In [153] a simple Fuzzy ART neural network approach is presented. It classifies and compresses observed continuous signals at the sensor nodes. The compressed output is then sent to a base station, where system-wide event classification can be performed. This has been implemented in an office monitoring application [154] and is presented in Chapter 9.

## 8.1 Introduction

Event classification and anomaly detection have gained much attention in wireless sensor networks as presented in Section 3.6. The proposed approaches cover a considerable range of applications. Nevertheless, simple, lightweight, but still self-learning algorithms that are based on the information collected in tracking groups have rarely been considered. In particular, an energy-efficient system for the detection of illegal building intrusion by wireless sensor networks has not yet been provided. Many existing approaches either do not provide sufficient accuracy or impose too high demands in terms of communication and storage to address the problem. In particular, many systems are tailored to highly accurate, short-term deployments. Considering building monitoring such an approach is not applicable, though. In our application a physical environment has to be permanently monitored. Consequently, frequently loading batteries is not feasible. On the other hand, simple threshold-based detection systems are not flexible and require expert knowledge to determine best-suited thresholds. For these reasons we have developed problem-specific lightweight solutions that run on top of the networking functionality provided by DELTA.

Our system provides both, functionality to classify time-discrete events and

means to distinguish abnormal from normal behavior. Both problems impose different demands on the respective system.

The classification of time-discrete events is addressed in Section 8.2. Thereby, the content of event reports received from DELTA group leaders is classified according to event classes learned from training data. A classifier based on Fuzzy Logic is proposed. It is compared to a probabilistic classifier and a neural network approach. The trade-off between classification accuracy (false alarm prevention) and reporting latency is investigated. In the evaluation five different light bulbs are classified according to their estimated signal strength emission (see the implementation on TmoteSky nodes Section 7.3.2). Light sources are classified according to their emitted photosynthetic active radiation (PAR) and total solar radiation (TSR) values. With the proposed classifier, classes of event that are a priori known can be classified in an unsupervised way.

To deal with continuous events that evolve randomly in and over time, the Fuzzy Logic Controller presented in Section 8.2 is not convenient. An accurate classification of continuous event patterns with respect to previously learned event classes is barely implementable due to resource constraints in distributed wireless sensor networks if long-term operation is required. This is mainly because storage capacity is limited and because arbitrary, randomly occurring events are difficult to predict. On the other hand, if the classification of events is not necessary, but the detection of anomalies is sufficient, the resulting storage complexity can be lowered. This allows the implementation of anomaly detection functionality in wireless sensor networks. An adequate anomaly detector on node level is proposed in Section 8.3. The algorithm is based on Adaptive Resonance Theory (ART). An ART neural network is a simple kind of neural network which represents an adaptive memory. Thus, a given sensor is able to learn some (local) prototypes of event patterns. With this adaptive memory each sensor node is able to report (temporary) unknown event patterns. Our ART neural networks are extended with an aging mechanism, which allows discarding old, rarely recognized event patterns to make space for new unknown event patterns. Therefore, our ART neural networks provide the sensor nodes with a short term memory. Thus, the learning capability can be maintained. The ART neural networks have been used to detect and report flashlight periods. The TmoteSky sensor nodes measuring PAR have been used. Hence, the ART neural networks process time vectors $\mathbf{z} = \{z_1, ..., z_N\}$, which describe the evolution of the PAR light signal over a specific interval. The anomaly detector is completely self-learning, adaptive, and does not require any a priori knowledge of the kinds of events that will occur. Drawbacks are poorer accuracy compared to classifiers that are tuned from training data.

In Section 8.3 the application of ART neural networks for local detection and reporting of flashlight periods is investigated. In subsequent work, ART neural networks have been used for building monitoring. Our office monitoring system based on ART neural networks is presented in Chapter 9. For office monitoring, passive infrared (PIR) and vibration sensors have been used.

## 8.2 Classification and Filtering of Time-Discrete Events

In this section the classification of time-discrete events computed on TmoteSky nodes is investigated [152]. Three different classifiers are evaluated: a Bayesian classifier, our developed Fuzzy Logic Controller (FLC), and a neural network approach. Generic monitoring applications pose an important requirement on classifier design. A priori knowledge of event classes is difficult to obtain. Hence, events are only observable as collections of raw sensor data. Consequently, event classes need to be learned from that raw (training) data. As a consequence, prelabeling of the events is not possible. In our work, event classes are learned by a k-means clustering algorithm (see Section 3.6.1). Any subsequent classifier training is based on these extracted event classes. Thus, the resulting classifiers are completely self-learning. Event classes are modeled according to training sets of emitted signal strength estimations collected with DELTA (see Section 7.3.2). The resulting event estimates are reported to a base station, where the classifiers are trained. The learned classifier parameters are then downloaded onto the sensor nodes, where any subsequent classification and filtering is performed.

### 8.2.1 Introduction

Environmental data is sensed and processed on the sensor nodes. As discussed before, event reports are commonly sent to a base station, where the data is stored and/or further processed. Reporting the measurements from all sensor nodes to the base station is expensive. Therefore, the DELTA algorithm has been developed. With DELTA events are monitored in a distributed manner and only the computed estimates are reported to the base station. In order to allow an event detection system to respond to collected event reports, classification functionality is required. Classifiers might be implemented at the base station or even on sensor node level. By implementing the classifiers on node level, false or non-relevant event reports can be filtered in the network. Thus, communication costs can be saved. The classification and filtering of event reports requires means to distinguish different event types on one hand and confidences in the classifications on the other. For anomaly detection a distinction between different event types is not necessary. Thus, our anomaly detection algorithm proposed in Section 8.3 follows another approach.

In our work we consider classifiers which are modeled based on measured data. Therefore, the event classes and the parameters of the classifier are learned, respectively estimated, from training data. In many applications such as most monitoring systems, it would be difficult to predetermine event classes. Moreover detailed expert knowledge would be needed. To avoid these drawbacks unsupervised learning techniques have been developed. Relevant related work is presented in Section 3.6. Learning from data has the advantage that no expert knowledge is required. Thus, the application of the algorithm is simpler and design flaws due to poor data abstractions can be prevented.

Another classifier requirement is the ability to filter suspicious data (e.g., false

alarms). To support this task, a classifier based on fuzzy logic concepts has been developed. A neural network approach that provides the same functionality has been evaluated too. Both classifiers assign a confidence degree to every classification. Thus, it is possible to filter time-discrete events that do not satisfy a given threshold requirement. Because non-satisfying event reports are filtered, a periodic reclassification of events is required. In most applications such as building monitoring or event tracking, a continuous monitoring with periodic re-computations of the event characteristics is needed per default. Obviously, the filtering of event reports has an impact on reporting delays. If more event reports are filtered, fewer reports arrive at the base station. This might introduce additional delays until an event is reported. On the other hand, if reports are processed unfiltered, false alarms might be reported and subsequent actions might arise, which implies energy costs and might lead to financial costs. Therefore, the trade-off between false-alarm tolerance and reporting latency is investigated.

In contrast to classification, which can easily be executed on sensor nodes, the estimation of the classifier parameters, i.e., the classifier tuning, is performed at a base station. This has two reasons: First, the classifier is tuned from training data which needs to be collected at a central instance anyway. Second, the tuning of the classifier is too expensive to be run on a simple sensor node, but can be performed at a base station equipped with more computation power. Alternatively, if the event reports are routed to a dedicated station in the Internet, the classifier tuning can be performed there, too. Once the parameters of the classifier have been computed, they only need to be downloaded onto the sensor nodes, which perform any subsequent classification.

### 8.2.2 Fuzzy Logic Controller

In this section our Fuzzy Logic Controller (FLC) is presented. The basic concepts of fuzzy logics have been presented in Section 3.6.3. All computations are based on the clusters learned by fuzzy k-means (see Section 3.6.1). We assume that $m$ different event classes $C_i$ have been extracted. Fuzzy Logic Controllers classify event patterns according to classification rules. Accordingly, the premises and consequences of the rules need to be determined. Common Fuzzy Logic Controllers model the premises and consequences based on expert knowledge. This is appropriate for systems where the expected event types can be predicted. Because this is difficult in event detection and monitoring systems, we model the classification rules from training data. In Section 3.6.3 Takagi Sugeno TSK classifiers have been presented that provide the desired functionality. TSK classifiers have the additional desired property that each conclusion of a classification rule assigns a given event pattern with a certain confidence to the specific class the pattern is expected to belong to. This allows the assignment of belief degrees to classifications, which false-alarm filtering. Therefore, we have decided to model our Fuzzy Logic controller according to the TSK2 classifier model (see Section 3.6.3).

In the following we show how the tuning of both the premises and the conse-

quences is performed in our approach. The idea is to generate exactly one rule for each event class (represented as cluster). Accordingly, each classification rule is used to classify samples of one specific event class.

### Tuning the premises

The premises are tuned from the clusters learned by fuzzy k-means. Thereby, the fuzzy sets of the premises are the projections of the clusters on the coordinate axes, where the cluster center has highest membership degree. The cluster, actually a scatterplot of event samples (estimations) belonging together, is visualized by a bell in Figure 8.1.

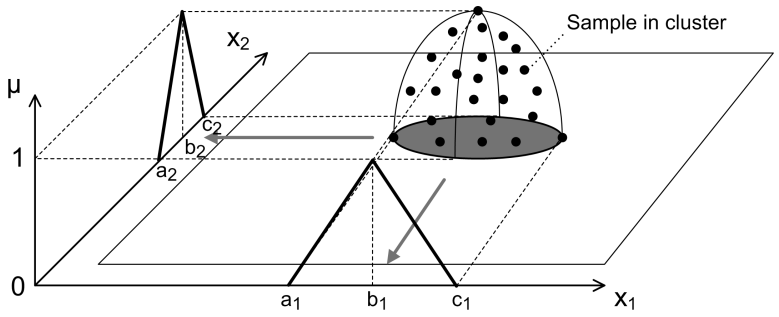



**Figure 8.1:** Mapping a cluster to the fuzzy sets $\tilde{\mathbf{A}}_{\mathbf{k,i}}$ of the premises in $\Re^{\mathbf{2}}$.

The projection of a 2-dimensional cluster, i.e., the samples consists of two kinds of phenomena, is shown in Figure 8.1. In Figure 8.1 the cluster is modeled according to triangular membership functions. The rule $R_k$ used to classify samples into the cluster above looks as follows:

$$R_k : IF\ \mu_{\tilde{A}_{k,1}}(x_1) \wedge \mu_{\tilde{A}_{k,2}}(x_2)\ THEN\ g_k^{TSK2}(\mathbf{x}) \tag{8.1}$$

Different kinds of membership functions $\mu_{\tilde{A}_{k,i}}$, with $i = 1, .., n$, can be extracted from a cluster. In our work triangular and Gaussian membership functions are used. In order to parameterize a triangular function $\mu_{\tilde{A}_{k,i}}$, the minimum, mean, and maximum values of a cluster in dimension i are needed. The determination of the parameters of a Gaussian membership function $\mu_{\tilde{A}_{k,i}}$ requires the mean and the standard deviation of a cluster in dimension i. The projections are approximations and do not necessarily model the samples in the clusters exactly.

### Tuning the consequences

In order to model the functions $g_k^{TSK2}$ of the consequences (see Section 3.6.3), the estimates $z_{k,i}$ have to be computed (see Equation 3.14). These estimates are computed for each rule $R_k$. Therefore, the degree of satisfaction (significance) of

the premise of each rule $R_k$, in the following labeled as firing strengths $\tau_k(\mathbf{z})$, needs to be determined (consider also the premise in rule (8.1)):

$$\tau_k(\mathbf{z}) = \mu_{\tilde{A}_{k,1}}(\mathbf{z}_1) \wedge \mu_{\tilde{A}_{k,2}}(\mathbf{z}_2) \wedge ... \wedge \mu_{\tilde{A}_{k,n}}(\mathbf{z}_n) \tag{8.2}$$

For the TSK2 classifier (see Equation (3.14)) the product is used as aggregation function $\wedge$. In order to obtain the firing strength $\beta_{k,i}$, of a rule $R_k$ to a cluster $C_i$, the sum of firing strengths $\beta_1, ..., \beta_m$ of all samples belonging to that specific cluster $C_i$ are computed according to [72]:

$$\beta_{k,i} = \sum_{\mathbf{z} \in \mathbf{Z}} Ind(\mathbf{z}, C_i)\tau_k(\mathbf{z}), \qquad \forall k = 1, ..., m; \tag{8.3}$$

where $Ind(\mathbf{z}, C_i)$ indicates the cluster $C_i$ to which the element $\mathbf{z}$ belongs:

$$Ind(\mathbf{z}, C_i) = \begin{cases} 1, & \text{if } \tau_k(\mathbf{z}) \text{ is maximum,} \\ 0, & \text{else.} \end{cases} \tag{8.4}$$

This means that after having determined the cluster $C_i$ with maximum firing strength as 'class label' of a sample $\mathbf{z}$, the firing strengths of $\mathbf{z}$ with respect to each rule $R_k$ is assigned to the according sum in $\beta_{k,i}$. An example illustrates the procedure. Given the set of samples $\mathbf{Z} = \{\mathbf{z}_1, ..., \mathbf{z}_4\}$ and four rules $R_1, ..., R_4$, one for each cluster $C_1, ..., C_4$, the firing strengths $\tau_1, ..., \tau_4$ of each sample $\mathbf{z}$ to each rule are:

| Sample | $\tau_1(\mathbf{z})$ | $\tau_2(\mathbf{z})$ | $\tau_3(\mathbf{z})$ | $\tau_4(\mathbf{z})$ | Class |
|---|---|---|---|---|---|
| $\mathbf{z}_1$ | **0.6** | 0.2 | 0.1 | 0.1 | $C_1$ |
| $\mathbf{z}_2$ | 0.1 | **0.7** | 0.1 | 0.1 | $C_2$ |
| $\mathbf{z}_3$ | **0.4** | 0.3 | 0.1 | 0.2 | $C_1$ |
| $\mathbf{z}_4$ | 0.2 | 0.2 | **0.5** | 0.1 | $C_3$ |

Sample $\mathbf{z}_1$, for example, has a firing strength of 0.6 in respect to $C_1$, of 0.2 in respect to $C_2$, and of 0.1 in respect to $C_3$ and $C_4$, respectively. According to Equation (8.4), $\mathbf{z}_1$ is therefore assigned to $C_1$ (bold in Table 8.2.2). Similar assignments are obtained for the other samples. By applying Equation (8.3) the matrix $\beta_{k,i}$ is filled as follows:

$$\beta_{k,i} = \begin{bmatrix} 0.6 + 0.4 & 0.1 & 0.2 & 0 \\ 0.2 + 0.3 & 0.7 & 0.2 & 0 \\ 0.1 + 0.1 & 0.1 & 0.5 & 0 \\ 0.1 + 0.2 & 0.1 & 0.1 & 0 \end{bmatrix}$$

In the first column $\beta_{k,1}$ of $\beta_{k,i}$, the firing strengths of $\{\mathbf{z}_1, \mathbf{z}_3\}$, which belong to cluster $C_1$, are recorded in respect to the according rule $R_k$, i.e., to the respective row in $\beta_{k,1}$. In the second column the firing strengths of $\mathbf{z}_2$, which belongs to $C_2$,

are recorded. The fourth column of $\beta_{k,i}$ contains no elements as no sample falls in the according cluster $C_4$. The $z_{k,i}$ are finally obtained by normalizing the $\beta_{k,i}$:

$$z_{k,i} = \frac{\beta_{k,i}}{\sum_{s=1}^{m} \beta_{k,s}} \tag{8.5}$$

Both, premises and consequences are tuned at a management station, e.g., a base station, where the training data has been collected. The estimated membership functions $\mu_{\tilde{A}_{k,i}}$ and the matrix $z_{k,i}$ are the only information which have to be downloaded onto the sensor nodes. Any subsequent classification of input $\mathbf{x} \in \Re^n$ can then be performed on the sensor nodes by applying Equation (3.14) .

### 8.2.3 Classifier Configuration

In the following the classifier configurations are presented. In addition to the FLC, a simple Bayesian classifier (see Section 3.6.2) and a FeedForward Neural Network approach (FFNN) (see Section 3.6.4) have been evaluated. All three classifiers have been tuned with the same training sets. For the evaluation the same test sets have been used.

The data from the signal strength estimation experiments on the TmoteSky nodes is classified (see Section 7.3.2), i.e., five different light bulbs are classified according to their emitted Photosynthetic Active Radiation (PAR) and Total Solar Radiation (TSR). Hence, each pattern $\mathbf{x}$ consists of two values that are fed into the respective classifier. The feature or phenomena space consequently is $\Re^2$.

The FLC classifier has been evaluated with both Gaussian and triangular membership functions. Gaussian functions have the general advantage that the whole feature space is covered. This is a nice feature, which has little impact in our kind of application, though. If a pattern cannot be assigned to a given cluster, the sample is just not classifiable with respect to the given event classes ($\mu_{\tilde{A}} = 0$). This is a reasonable property in our context. Hence, the triangular membership function introduces no drawback in that respect.

The FFNN running on the sensor nodes has been implemented with 10 hidden neurons. Thus, storage complexity of the FFNN is similar to that of the FLC. The weights of the FFNN have been computed at the base station from the training set. For classification, only the weights need to be downloaded onto the sensor nodes. Like the FLC method the FFNN method assigns a confidence to its classifications.

Considering the Bayesian classifier, the different $\mathbf{K}_i$ and $\mathbf{m}_i$ are again estimated at the base station and then downloaded onto the sensor nodes. Thereafter, classifications can be performed on a sensor node according to rule (3.7) in Section 3.6.2. The Bayesian classifier does not support any confidence in its classifications. A sample is fixedly assigned to the cluster it belongs to with highest probability.

Before providing classification results of the FLC classifier in Section 8.4, the implementation of ART neural networks for local signal processing and anomaly detection is presented in the next section.

## 8.3 Local Signal Processing with ART Neural Networks

The previous chapter has introduced techniques to classify time-discrete events. The assumption was made that the expected time-discrete events are either known or occur frequently. Thus, expected classes of events can be learned in specific learning periods. On the other hand, applications such as building monitoring require means to distinguish abnormal from normal behavior. Normal behavior can be learned to some extent. However, learning abnormal behavior is much more difficult and sometimes even impossible. Since modeling unexpected (abnormal) behavior is commonly not possible and an overall knowledge of expected (normal) behavior is both difficult to determine and storage-consuming, the classification methods proposed in the last chapter are not tailored to to anomaly detection.

### 8.3.1 Introduction

The goal of this implementation has been to detect and report abnormal light sensor activation patterns (flashlight periods) on a sensor node. Previously, the DELTA algorithm was used to detect and track persons equipped with a flashlight. The goal of this work was to learn and report given light sensor activation patterns. The light sensor activation patterns are composed of signals, i.e., collected time series of light measurements $\mathbf{x} = \{x_1, ..., x_N\}$, which are monitored on the sensor nodes. The classification problem faces mainly two restrictions in sensor networks. First, processing power and memory are limited on sensor nodes. Consequently, complex pattern classification methods are difficult to be implemented on node level. On the other hand, communication costs are high. Therefore, it is not possible to transmit the observed signals unprocessed to a fusion center.

Therefore, we propose an adaptive memory approach that provides online learning capability on node level. Observed measurement series, periodically collected in predefined intervals, are processed. Consider the following series $\mathbf{x}$ = [0, 0, 123, 64, 111, 0, 0, 0, 3, 0] of light measurements in Lux sampled by a node in two seconds. By applying the adaptive memory, $\mathbf{x}$ is classified and compressed with respect to the current memory of the node. This means, if $\mathbf{x}$ is known, the classification number is returned by the adaptive memory. Otherwise, -1 which represents a not classifiable or unknown event is reported. Only this classification output is transmitted to a central fusion center, e.g., a DELTA leader node or a base station. Thus, event patterns are classified and compressed at node level, which reduces the amount of reporting data considerably. At the fusion center, the system-wide classification, of the different local decisions, received from the sensor nodes, is performed. In this section the decision unit which classifies and compresses observed signals on node-level is introduced [153]. The system-wide decision unit is presented in Chapter 9.

In our work, the adaptive memory is implemented as a simple neural network based on Adaptive Resonance Theory (ART) (see Section 3.6.5). Our Fuzzy ART neural network learns and classifies sequentially present analog input vectors. The

number of stored event classes (prototypes) can be controlled and no buffering of input vectors is necessary. Moreover, the sequential processing and learning imposes moderate computational complexity, which makes the algorithm applicable to sensor nodes. Any observed input is compared and classified with respect to a maximum number of $M$ prototypes, which are learned and continuously updated by the Fuzzy ART. Considering $M$ prototypes and an input vector of size $N$ the algorithmic complexity, both in time and memory usage, has been shown to be in the order of $O(MN)$ [15]. The theory of Fuzzy ART neural networks was discussed in Section 3.6.5. Some adaptations to meet the requirements of wireless sensor nodes are presented in Section 8.3.2. Real-world experiment results are provided in Section 8.4.2.

### 8.3.2 Local Fuzzy ART Neural Network

In Chapter 6 we have introduced the DELTA tracking algorithm. Delta was used for terrain monitoring during night. Persons equipped with flashlights were tracked. Hence, flashlight periods need to be distinguished from normal illumination patterns. This can be well done with ART neural networks. Advantage of the ART neural networks is their unsupervised nature and their efficiency in terms of memory usage and computation costs.

Traditional ART neural networks provide a long-term memory of $M$ categories. When all $M$ neurons in the comparison layer F2 of the ART neural network are used, the learning capability of the ART neural network is exhausted. Any new input pattern, even if it occurs frequently, can no longer be learned. Hence, 1 (unknown) would be returned for every such input pattern. On the other hand, we envision a mechanism that recognized frequently present input patterns, while sporadic input patterns shall be classified as unknown. Therefore, we have changed the common ART neural network design to implement short-term memory. Short term memory is implemented by an aging mechanism. After every monitoring cycle, the age of each prototype that has not been matched is incremented. Thus, sporadically matched prototypes become older quickly. As soon as the memory of the ART neural network is full, always the oldest prototype in the memory is replaced by the current, unknown input pattern. The approach is reasonable as frequently matched prototypes (normal input) are hardly affected by the aging mechanism. Thus, frequent input is recognized and filtered by the ART neural network.

Finally, traditional ART neural networks return the category number if a category is determined for a given input time vector $\mathbf{x}$ and -1 otherwise. In contrast, our ART-based event detector returns 1 (state has changed) if the classification number of two subsequent classifications has changed and 0 (no state change) otherwise. Flashlight periods lead to many classification state changes. Hence, the number of classification state changes is used as indicator for flashlight periods. The ART neural networks used in the detection of flashlight periods provide the following features:

- The ART neural networks are very lightweight.

- Repeated illumination conditions are filtered by the ART neural network. This saves communication costs.
- Flashlight periods lead to many classification state changes and thus to reports.
- A data compression of $N$ to 1 is achieved, which minimized data volume.

In the implementation light signals are continuously monitored in intervals of 2 s. Within this 2 s the light intensity in Lux is sampled 40 times. To decrease the size of these time series, two discrete Haar Wavelet transforms are applied (see Section 3.6.6). The resulting analog input time vector $\mathbf{x}$ of size 10 is fed to the Fuzzy ART network. The input time vector $\mathbf{x}$ represents the monitored light signal over two seconds and might for example look as follows: [0 0 0 388 298 0 0 0 0 0] in Lux. In this example the light sensor would have measured light in the second half of the first monitoring second. This pattern is then classified in dependence of the stored prototypes in the memory of the Fuzzy ART neural network.

The recognition (comparison) layer F2 of the Fuzzy ART neural network allocates memory for 10 categories. This means, the adaptive memory implemented on the sensor nodes can store up to ten event pattern prototypes. The resulting storage requirements are in the order of 200 bytes for the Fuzzy ART neural network (each weight requires 2 bytes). The Fuzzy ART neural network requires only comparisons and simple arithmetic operations. To save communication costs, twenty subsequent Fuzzy ART outputs are stored into a single report vector that is sent to a fusion center every 40 s. Thus, 800 samples collected over a period of 40 s are compressed to 20 values. The sensitivity threshold of the Fuzzy ART neural network $\rho$ is 0.75 and the learning rate is 0.1.

## 8.4 Evaluation

In the following two sections the classification performance of the Fuzzy Logic Controller as well as the local signal processing performance of the Fuzzy ART neural network are shown.

### 8.4.1 Classification of Time-Discrete Events

The configuration of the classifiers was presented in Section 8.2.3. TmoteSky nodes (see Section 3.8.2) were used. Five light bulbs of 25, 40, 60, 75 and 100 Watt were classified according to their emitted PAR and TSR values. The experiment is depicted in Figure 7.6. The required training and test sets were generated as follows: the emitted PAR and TSR values were estimated 30 times for each bulb at 15 event locations. The resulting training and test sets consist of 2050 event patterns, consisting of PAR and TSR light emission estimates. The mean emitted signal strength errors $\Theta$ and the standard deviations $\sigma$ in Lux of the sets computed on the TmoteSky nodes are provided in Table 7.3 in Chapter 7. These values only indicate the distribution of the patterns in the sets. For classification neither the mean signal strength errors $\Theta$ nor the standard deviations $\sigma$ are needed. Two test sets, namely a training set and a test set have been computed. The training set was used at the base station to tune the classifiers, while the patterns in the test set were classified on a sensor node to evaluate the performance of the respective classifier.

#### False-Alarm Filtering

The FLC and the FFNN provide a confidence in their classifications. The confidence is a value between 0 and 1. A value of 1 means the classifier has perfect confidence in its classifications. The filtering of events is done by rejecting classifications that do not satisfy a given confidence requirement $T_\mu$. Table 8.1 shows the performance of the different classifiers based on the classification error rate, i.e., of the rate of wrong classifications (false positives) in dependence of $T_\mu$.

All classifiers perform almost equally well if no filtering is applied ($T_\mu = 0$). Thus, a classification error-rate of approximately $10\%$ can be achieved. With FLC and FFNN arbitrary classification error-rate constraints can be satisfied. If, for example, the classification error rate needs to be smaller than $5\%$ (marked bold in Table 8.1), the FLC-Gaussian and the FFNN require a $T_\mu$ of 0.5, while FLC-Triangular requires a $T_\mu$ of 0.4. The Bayesian classifier does not support filtering, thus its classification error-rate remains constant.

#### Outlier Filtering and Reporting Latency

The FLC and the FFNN filter events that do not satisfy a given $T_\mu$. As a consequence some reporting delays are introduced. As mentioned in Section 7.3.2, classification is performed at the leader node every 0.5 s. The classification is aborted in the current implementation if $T_\mu$ is not satisfied after 30 computations, i.e., if the

**Table 8.1:** Error rates of the classifiers.

| | Bayesian | FLC-Gaussian | FLC-Triangular | FFNN |
|---|---|---|---|---|
| $T_\mu$ | Error [%] | Error [%] | Error [%] | Error [%] |
| 0 | 9.86 | 9.56 | 10.64 | 11.07 |
| 0.1 | | 9.01 | 9.68 | 9.22 |
| 0.2 | | 8.56 | 9.07 | 7.45 |
| 0.3 | | 7.03 | 6.23 | 5.92 |
| 0.4 | | 6.35 | **4.6** | 5.02 |
| 0.5 | | **4.85** | 2.95 | **3.34** |
| 0.6 | | 3.42 | 2.61 | 1.95 |
| 0.7 | | 3.34 | 2.37 | 0.76 |
| 0.8 | | 3.64 | 0 | 0.83 |
| 0.9 | | 2.9 | | 0 |

reporting time exceeds 15 seconds. To evaluate the reporting delay five characteristic distribution parameters (in seconds) have been computed: The minimum and maximum, the median, and the lower and upper quartiles.

**Table 8.2:** Reporting delays of triangular FLC.

| | Error rate | Latency percentiles p in [s] | | | | |
|---|---|---|---|---|---|---|
| $T_\mu$ | [%] | Min | 25-p | Median | 75-p | Max |
| 0 | 10.64 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| 0.1 | 9.68 | 0.5 | 0.5 | 0.5 | 0.5 | 3.5 |
| 0.2 | 9.07 | 0.5 | 0.5 | 0.5 | 1 | 10 |
| 0.3 | 6.23 | 0.5 | 0.5 | 0.5 | 1 | $> \mathbf{15}$ |
| 0.4 | **4.6** | **0.5** | **0.5** | **1** | **6** | |
| 0.5 | 2.95 | 0.5 | 0.5 | 1.5 | 7.5 | |
| 0.6 | 2.61 | 0.5 | 1 | 4.25 | 13.5 | |
| 0.7 | 2.37 | 0.5 | 2.5 | 10.5 | $> 15$ | |
| 0.8 | 0 | 0.5 | 4.5 | $> 15$ | | |
| 0.9 | | 0.5 | $> 15$ | | | |

Table 8.2 shows the reporting latencies if applying the triangular FLC classifier with varying $T_\mu$. The minimum reporting delays to achieve an error-rate below $5\%$ are marked bold. The results show that in $50\%$ of all experiments (median) event reports are generated within 1 s. Accordingly, the classification needs to be performed only twice to generate an event report. In $75\%$ of all experiments a report is sent within 6 s. At least in one experiment no report could be generated within 15 s. As expected, if $T_\mu$ is increased, the reporting delays increase too.

**Table 8.3:** Reporting delays of FFNN.

| | Error rate | Latency percentiles p in [s] | | | | |
|---|---|---|---|---|---|---|
| $T_\mu$ | [%] | Min | 25-p | Median | 75-p | Max |
| 0 | 11.07 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| 0.1 | 9.22 | 0.5 | 0.5 | 0.5 | 0.5 | 1 |
| 0.2 | 7.45 | 0.5 | 0.5 | 0.5 | 0.5 | 2.5 |
| 0.3 | 5.92 | 0.5 | 0.5 | 0.5 | 0.5 | 2.5 |
| 0.4 | 5.02 | 0.5 | 0.5 | 0.5 | 1 | 2.5 |
| 0.5 | **3.34** | **0.5** | **0.5** | **0.5** | **1** | **4** |
| 0.6 | 1.95 | 0.5 | 0.5 | 1 | 1.5 | $> 15$ |
| 0.7 | 0.76 | 0.5 | 0.5 | 1 | 1.5 | |
| 0.8 | 0.83 | 0.5 | 1.5 | 1.5 | 3 | |
| 0.9 | 0 | 0.5 | 1.5 | 3 | 7 | |

The FFNN outperforms the FLC in terms of reporting delay (see Table 8.3). Considering an error-rate lower than 5%, in 50% of all experiments (median) a correct event report is immediately sent. Moreover, even in the worst case a report is reported within 4 seconds. This is because the neural network is able to model the distribution of the event patterns better than the FLC, which needs the abstraction of triangular or Gaussian distributions. On the other hand, the knowledge processing is hidden by the FFNN. This makes it more difficult to understand and configure the classifier. If the initial values are poorly chosen, the FFNN can perform much worse than the FLC.

### 8.4.2 Anomaly Detection and Signal Processing on Node Level

In this section the performance of a Fuzzy ART neural network for signal compression and classification on single sensor nodes is investigated. The TmoteSky platform was used (see Section 3.8.2). The goal is to detect and distinguish flashlight periods from normal illumination patterns. The Photosynthetic Active Radiation (PAR) sensor has been used. The ART neural network configuration has been presented in Section 8.3.2.

All experiments started during daytime in a bright lecture room. Two different light patterns were tested. The first light pattern is shown in the upper part in Figure 8.2. First, the blinds were lowered. Then, the blinds were raised and lowered again, before two flashlight periods occurred. In the second light pattern (in the lower part in Figure 8.2) the raising and lowering of the blinds was substituted with turning the room illumination on and off. Moreover, an additional room illumination on/off period was performed between the two flashlight periods. Both experiments were repeated three times.

In both patterns two flashlight periods were included. In these periods, a person

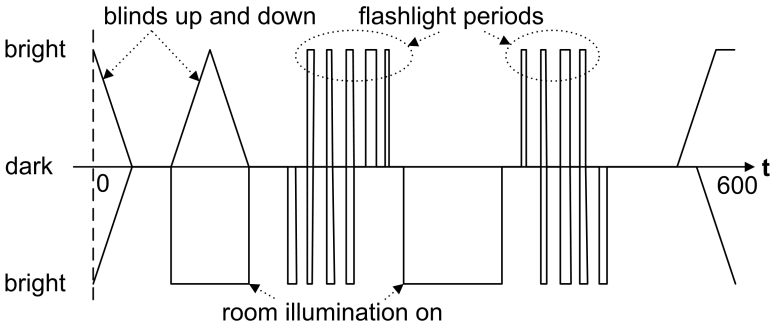



**Figure 8.2:** Light pattern I (upper part) and light pattern II (lower part).

equipped with a flashlight entered the room and searched the room with the flashlight. Thereby, the sensor node was illuminated from time to time depending on the searching. Accordingly, the flashlight period invoked a temporarily increased number of bright/dark switches on the sensor node. Instead of the classification numbers, changes in the classifications (state changes) are reported.

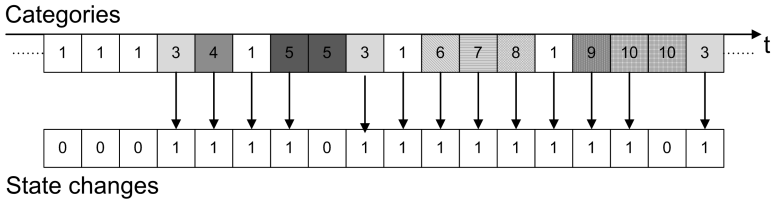



**Figure 8.3:** Sequence of changes of classification outputs.

An example of a flashlight period is shown in Figure 8.3. In the beginning event patterns of prototype 1 were detected for a couple of times. Thereafter, different prototypes were detected in a short time. This is due to the many bright/dark switches imposed by the flashlight period. The changes in classifications represent the frequency in presence of different input patterns. Therefore, this measure was used to detect flashlight periods. In office monitoring (see Chapter 9) we followed another approach. It is important to remember that only state changes indicated as 1 are reported by the observing sensor node. Thus, considerable communication can be saved, since often classifications will not change, e.g., if no flashlight is present during night time. Moreover, the mechanism is completely self-learining and adaptive.

The results with the first light pattern are shown in Figure 8.4. In all three experiments the lowering and raising of the sun-blinds did not lead to any state changes. Accordingly, the Fuzzy ART anomaly detector is able to adapt to slowly changing environmental conditions. The adaptation behavior depends on parameterization of the Fuzzy ART network, though. Both flashlight periods are easily identifiable by the respective accumulations of state changes in Figure 8.4. In experiment 1, the

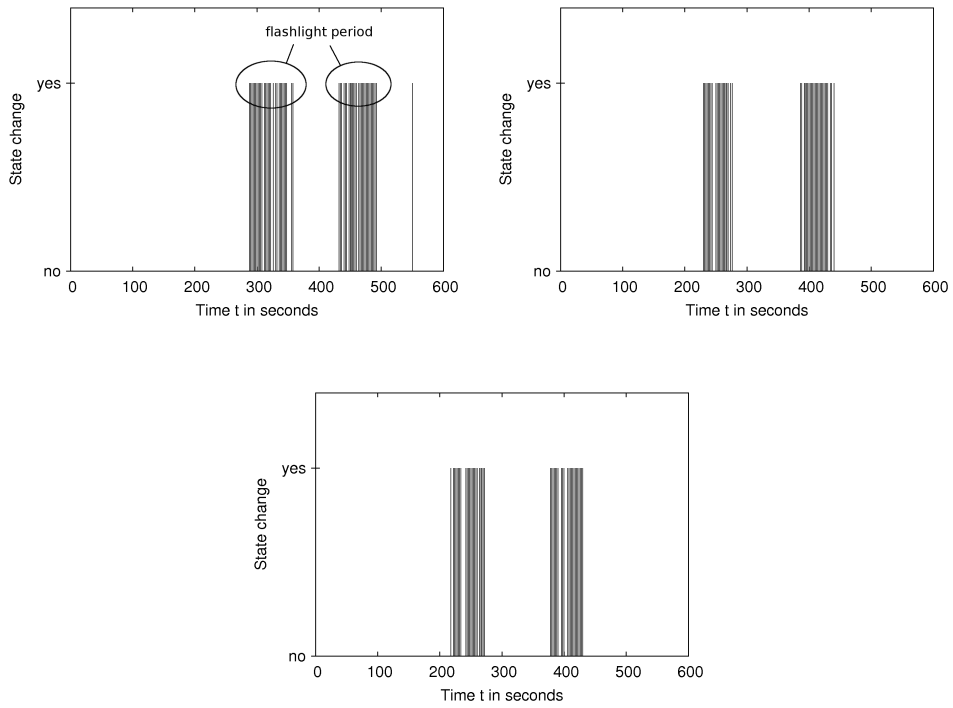



**Figure 8.4:** Occurrence of state changes in experiments 1 to 3.

raising of the sun-blinds in the end invoked one state change, which could easily be filtered by the system.

In contrast to experiments 1 to 3, experiments 4 to 6 contain two periods where the room illumination was turned on and off. This invoked few abrupt illumination changes shortly before and after the first flashlight period (see Figure 8.5). Considering the detection of the flashlight periods, the performance is similar to performance in experiments 1 to 3.

We conclude the evaluation by providing the mean number of state changes in all twelve flashlight periods which is 24.5, and the standard deviation which is in the order of 2. Accordingly, a person moving with a flashlight can well be distinguished from 'normal behavior' by observing at least a certain number of state changes in a specific interval. The filtering of subsequent event patterns has lead to communication savings of approximately 83%, compared to a mechanism that would have reported every ART output. A more detailed analysis of possible communication savings with ART neural networks is provided in the office monitoring application in the next chapter.

The algorithm is completely self-learning and does only require initial parameter settings for the number of allowed prototypes, the vigilance parameter and the learning rate. Depending on the observed signal and the memory constraints on the

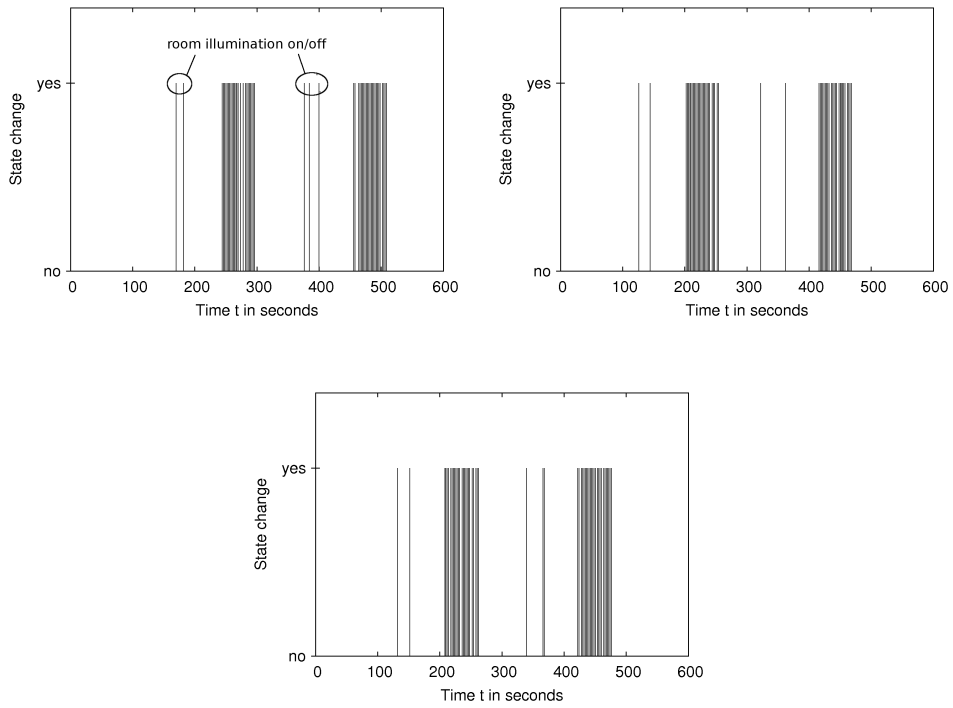



**Figure 8.5:** Occurrence of state changes in experiments 4 to 6.

sensor node these values might slightly differ.

## 8.5 Conclusions

In this chapter the classification of events has been addressed. Algorithms to classify time-discrete event patterns have been presented in Section 8.2. The trade-off between false-alarm prevention and reporting latency has been investigated. The evaluation has shown the trade-off between classification error-rate and reporting latency. Arbitrary error-rates have been achieved with FLC and FFNN. Error-rate minimization imposes reporting delays, though. If the application requires certain accuracy but can handle some delay, a neural network or a FLC classifier is an appropriate choice. The FFNN achieves slightly better latency results, but the tuning of the classifier is more critical. On the other hand, the FLC is easy to understand and performs well.

The Bayesian classifier has performed similar to FLC and FFNN without filtering. Even with moderate filtering, the Bayesian classifier is outperformed by the other solutions, though. All three solutions require training. Each trained classifier is performed at a leader node and requires only local neighborhood information. Thus, communication and computation overhead is kept small.

Considering the detection of unexpected, continuous events, the proposed time-discrete algorithms are not suited. This is mainly, because the algorithms require the modeling of event classes from training data. However, unexpected (abnormal) behavior can commonly not be predicted. Modeling everything that is expected (normal) is not feasible either. Therefore, we have implemented an adaptive memory approach based on Fuzzy ART neural networks that provides online learning and classification capability. Compared to the previous classifiers this approach achieves lower accuracy due to its sequential operation. However, storage costs are much smaller, which makes it a good choice for anomaly detection. The algorithm has been implemented to distinguish flashlight periods from normal room illumination patterns. The Fuzzy ART neural network has provided good performance in terms of detection accuracy and resource savings. Communication costs of approximately 83% could be saved.

In the next chapter an office monitoring system that uses ART neural networks to detect and report abnormal office occupancy is presented.

Chapter 9

# Office Monitoring Application

In the previous chapters we have presented the different modules of our event detection system. The general system architecture has been provided in Figure 1.1. A typical application scenario has been presented in Figure 1.2. In this chapter we present the deployment of the event detection system in an office monitoring application [154]. A common office room, offering space for two working persons, has been monitored with ten sensor nodes and a base station. The task of the system is to report suspicious office occupation such as office searching by thieves. On the other hand, normal office occupation should not throw alarms. In order to save energy for communication, the system provides all nodes with some adaptive short-term memory. Thus, a set of sensor activation patterns can be temporarily learned. The local memory is implemented as an Adaptive Resonance Theory (ART) neural network. Unknown event patterns detected on sensor node level are reported to the base station, where the system-wide anomaly detection is performed. The anomaly detector is lightweight and completely self-learning. The system can be run autonomously or it could be used as a triggering system to turn on an additional high-resolution system on demand. Our building monitoring system has proven to work reliably in different evaluated scenarios. Communication costs of up to 90% could be saved compared to a threshold-based approach without local memory.

## 9.1 Introduction

Subject of this chapter is the detection and reporting of abnormal behavior in building monitoring. Conventional building monitoring systems address the problem by deploying video surveillance systems (see Section 3.7.2). Such systems have a number of drawbacks, though. The system is expensive in terms of hardware, storage and communications. In particular if wireless technology is used, energy for communication becomes a critical issue. Furthermore, collecting multiple video streams imposes high demands on storage, online monitoring and video analysis. For example, the more video screens a security guard has to monitor, the higher is the probability that he misses some relevant information. Finally, a permanently active camera system is unpleasant for the office staff.

The envisioned system involves numerous research questions: is it possible to develop an energy-efficient reliable building surveillance system? Can the data volume be reduced while minimizing the probability of losing relevant information? Can privacy of staff be improved? Is the system economic?

Wireless sensor network technology provides means to develop such a system. Sensor networks are economic due to their lack of wires. Energy-efficiency is achieved by parallelism, distributed computing and in-network processing. Decisions can be made in the network, decreasing communication load and storage requirements. Hence, office occupation can be processed and filtered in the network. Because no pictures are taken, the identity of office staff can be hidden. For these reasons, we provide a wireless sensor network for building monitoring. The system can run standalone or it can be used to trigger a high-resolution system such as a (wireless) video surveillance system. A hybrid system would optimize accuracy, while keeping costs low. Unless something critical is detected by the sensor network, video surveillance is turned off. During this time, energy and storage costs are minimal and the identity of the office staff is concealed.

Tininess, resource constraints, need for long-term operation and dependency on batteries impose severe restrictions on wireless sensor networks. Hence, services provided in sensor networks need to be lightweight in terms of memory and processing power. Communication costs should be low. We satisfy these requirements by filtering normal (i.e., known) building occupation within the network. Normal behavior is temporarily learned by short term memory using an aging mechanism. The memory is implemented as Adaptive Resonance Theory (ART) neural network. ART neural networks compress observed event patterns to a single value representing the ART decision (known | unknown). Unknown patterns are reported to the base station. At the base station, the reported local decisions are fed to a binary ART neural network that performs the system-wide anomaly detection. ART neural networks are adaptive and learn sequentially. They require low storage and communication costs. These features are advantageous for use together with sensor nodes.

## 9.2 Office Monitoring

In this section the office monitoring application is introduced. We have used sensor nodes for a number of reasons. The system is lightweight, cost-effective, and easy to deploy. In particular no wires are required. The system conceals the identity of persons working in the office. Only sensor activation patterns can be determined. Thus, the system ensures privacy to the office staff. A high resolution system providing more detailed information could optionally be triggered. Thus, the application of the secondary, high resolution system could be restricted to periods of abnormal office occupation.

The deployment of the office monitoring system is depicted in Figure 9.1. In the current deployment the ART based anomaly detection software and the event

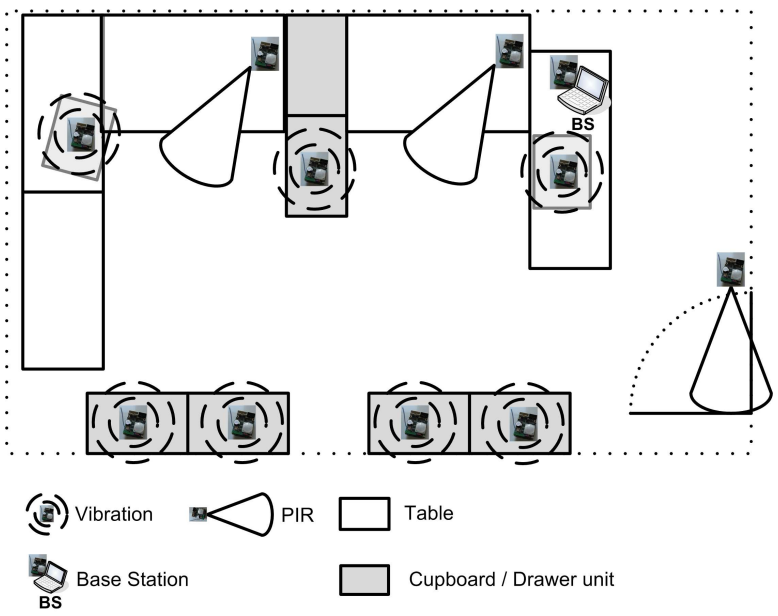



**Figure 9.1:** Office monitoring deployment.

detection and tracking functionality of DELTA were implemented. The medium access functionality has not yet been ported to the sensor nodes. A common office room of approximately 26.5 $m^2$ providing two working places was monitored. The room contains 5 office cabinets, 4 tables and two file cabinets. In total, 11 wireless sensor nodes were deployed: 3 nodes measuring Passive Infrared (PIR), 7 nodes measuring vibration and one node acting as gateway (base station). The 3 sensor nodes that measure PIR were placed such that the two working desks and the office entrance were monitored. The 7 vibration sensors monitored activation in the office cabinets and in the file cabinets. The sensor nodes do not report every sensor reading to the base station. This would be too communication intensive. Instead, series of sensor readings, 10 in the current implementation, are collected and processed on node level. The resulting local pattern classifications (known | unknown) are then sent to the base station, where the system-wide anomaly detection is performed.

### 9.2.1 System Design

Our building monitoring system can be summarized as follows. Series of sensor readings are periodically collected and processed on node level. In the current implementation the PIR and vibration sensors are processed every 2 s. Since series of ten sensor readings are processed, i.e., the monitoring resolution on node level is 20 s, time vectors $\mathbf{x} = \{x_1, ..., x_i; i = 10\}$ are locally processed by ART neural networks. As mentioned before, our ART neural networks implement an adaptive short term memory, which is based on an aging mechanism. The sensor nodes can store a certain number of prototype time vectors $\mathbf{y}$. Prototypes $\mathbf{y}$ that are not

matched by an input $\mathbf{x}$ become older. When the memory of the ART neural network is full, the oldest prototype $\mathbf{y}$ is replaced by the current input vector $\mathbf{x}$. With this mechanism learning can be continuously maintained. If $\mathbf{x}$ is not recognized in the ART memory, i.e., the input pattern is unknown, a 1 is signaled to the base station. Otherwise, no report is sent. It is not necessary to report locally known patterns, i.e., to signal a 0, because the base station expects the presence of a known time vector $\mathbf{x}$ if no report has been sent by the according node. Because each sensor node reports for every monitoring interval whether it has locally detected an unknown time vector $\mathbf{x}$ or not, the base station is able to perform global decisions based on the collected reports. Since ten sensor nodes are deployed, the base station processes space vectors of the form $\mathbf{z} = \{z_1, ..., z_j; j = 10\}$, where $z_i \in \{0, 1\}$ is the output of sensor node i.

An example may clarify the functionality. We assume, a person is searching two cabinet drawers that are equipped with vibration sensors 5 and 6. The first drawer is opened, then the second one, before the first drawer is closed, again followed by the second one. This is all done within a monitoring period of 20 s. Hence, time vector $\mathbf{x}_5$ containing the sensor readings of sensor node 5 might look like [0, 24, 12, 0 , 0, 0, 14, 22, 0, 0], while $\mathbf{x}_6$ is [0, 0, 0, 33, 0, 0, 0, 0, 41, 13]. Numbers of 0 mean no activation of the according sensor (PIR or vibration). Both vectors are locally processed by Fuzzy ART neural networks. We assume that $\mathbf{x}_5$ is recognized by the memory of sensor node 5, while $\mathbf{x}_6$ is not recognized by the memory of sensor node 6. Since Fuzzy ART neural networks store prototypes, this recognition behavior is possible. The other sensor nodes have not measured activation in the current monitoring period, i.e., they recognize time vector [0, 0, 0, 0, 0, 0, 0, 0, 0, 0] in their memory. In this example, only sensor node 6 reports a 1 (unknown time vector). Accordingly, the space vector $\mathbf{z}$ processed at the base station is [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]. The base station implements a binary ART neural network to process the binary input $\mathbf{z}$. The output of the binary ART at the base station is again 1 (unknown) if $\mathbf{z}$ is not recognized and 0 otherwise.

ART neural networks are predestinated to meet storage, local pattern recognition, and filtering requirements. The storage costs of an ART neural network are in the order of $O(MN)$, both in terms of time and memory (see Section 8.3). In the office monitoring application we have used the ESB sensor node platform (see Section 3.8.1). The available memory on these nodes for local signal processing is in the order of 300 bytes. Considering the available memory, the input vector size of 10, and 2 bytes to store floating points, 10 prototypes are stored in the memory of our ART neural networks. Finally, traditional ART neural networks return the category number if a category is determined for a given input $\vec{I}$ and -1 otherwise. In contrast, our ART-based event detector returns 0 (known) if $\vec{I}$ is recognized and 1 (unknown) otherwise. Hence, our ART neural networks provide the following features:

- ART neural networks are very lightweight.
- Known input vectors are filtered by the ART neural network, which saves

communication costs.

- A data compression of $N$ to 1 is achieved, which reduces data volume.

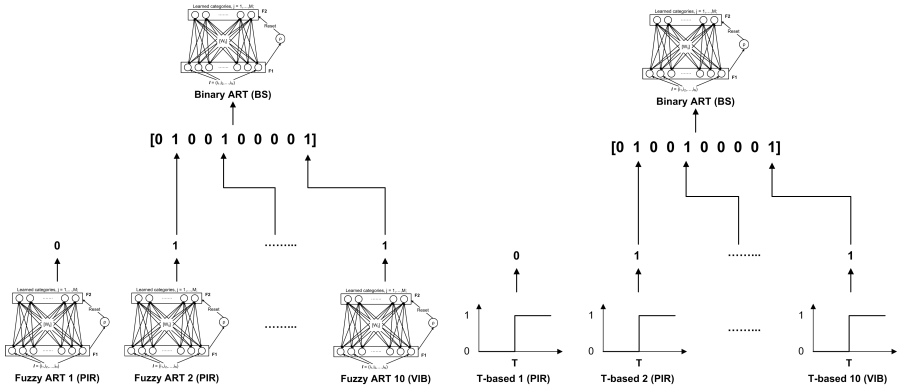
### 9.2.2 System-wide Anomaly Detection

In our basic system implementation, ART neural networks are implemented both at node level and at the base station. In related work thresholds are often used. Therefore, we also evaluated simple threshold-based decisions. In these cases, a 1 (unexpected input vector) is reported if the sum of $\mathbf{x}$ exceeds a predefined threshold T, i.e., if $\sum_{i=0}^{10} x_i > T$. Considering $\mathbf{x}_6$ = [0, 0, 0, 33, 0, 0, 0, 0, 41, 13] above, the sum is 87. Accordingly, if T is smaller than 87, a 1 is reported. Otherwise, no report is sent (the input is considered as expected or normal). The following four combinations are possible:

| Local | System-wide |
|---|---|
| Fuzzy ART | binary ART |
| ~~Fuzzy ART~~ | ~~Threshold-based~~ |
| Threshold-based | binary ART |
| ~~Threshold-based~~ | ~~Threshold-based~~ |

In the following evaluation the two crossed-out combinations are not considered: feeding local binary output to a threshold-based decider at the base station is not feasible. An example illustrates the corresponding problem. Consider two binary input vectors at the base station: $\mathbf{z}_1$ = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0] is an event that should be reported, while $\mathbf{z}_2$ = [0, 1, 1, 1, 0, 0, 0, 0, 0, 0] is an event that should be filtered. This potential case is not manageable with a threshold-based decider at the base station, because $\sum_{i=0}^{10} z_{1,i} = 1$ (should be signaled), while $\sum_{i=0}^{10} z_{2,i} = 3$ (should be filtered), which conflicts with a threshold-based decision. Next, we consider a system with threshold-based decisions on node level and at the base station. It is important that $\sum_{i=0}^{10} x_i$ is reported to the base station instead of 1 if $\sum_{i=0}^{10} x_i > T$. Otherwise, the problem could be reduced to the previous one. This completely threshold-based approach cannot be parameterized such that the resulting anomaly detection system works reliably.

The designs of the remaining two systems are depicted in Figure 9.2. In every monitoring period of 20 s, every sensor node j signals a locally unknown event pattern either if the current time vector $\mathbf{x}_j$ has not been recognized by the memory of node j (Fuzzy ART, see Figure 9.2(a)) or if $\sum_{i=0}^{10} x_i > T$ (threshold-based, see Figure 9.2(b)). Hence, the base station collects and processes space vectors $\mathbf{z} = \{z_1, ..., z_j; j = 10\}$, where j represents the ID of the respective sensor node and $z \in \{0, 1\}$. Thus, if an unknown event pattern (signaled as 1) has been reported by a specific sensor node j, neuron j in F1 of the binary ART neural network is activated, i.e., 1 is fed to the corresponding neuron. In Figure 9.2 this means that 1 is fed to neurons 2, 5 and 10 of the binary ART neural network. If the base station




**(a)** Local Fuzzy ART. **(b)** Local thresholds.

**Figure 9.2:** System designs with local memory (left) or local threshold (right).

did not receive a report from a sensor node, the base station assumes a recognized event pattern at the respective node. Hence, the corresponding neuron is fed with 0. This mechanism saves a considerable amount of reporting costs, because sensor nodes do not need to report known event patterns. The resulting input space vector $\mathbf{z}$ in Figure 9.2 is [0, 1, 0, 0, 1, 0, 0, 0, 0, 1]. In Figure 9.2 this input vector is not known by the binary ART memory and a system-wide unknown event is reported, i.e., the output of the binary ART is 1.

Since single system-wide event reports are not sufficient to accurately signal anomalies (office intrusions), a significance test determining accumulations of system-wide event reports is provided. This test evaluates the frequency of anomalies over a certain time period.

**Significance Test**

Significance $\Theta$ = 0;
Age of last unknown event $event\_age$ = 0;
**while** true
  Calculate binary ART ouput $\xi \in \{0, 1\}$;
  **if** $\xi$ == 1 // unknown event
    **if** $event\_age < T_{\text{max\_age}}$
      $\Theta$++;
      $event\_age$ = 0;
    **else**
      $\Theta$ = 0;
      $event\_age$ = 0;
    **end**
  **else** // known event
    $event\_age$++;
  **end**
  **if** $\Theta > T_{\text{significance}}$
    report anomaly;

The significance test is described in the pseudo-code above. The significance

test determines an alarm if in a certain time interval 5 system-wide unknown events ( $\Theta > T_{\text{significance}}$, $T_{\text{significance}}$ = 4) have been signaled by the binary ART neural network. Thereby, the time difference between every subsequent pair of events must be smaller than 80 s ($T_{\text{max\_age}}$ = 5, which implies 4 · 20 s).

Since five system-wide events are required, at maximum 320 s (4 · 80 s) can elapse until an office intrusion is reported. On the other hand, the minimum delay is 80 s if 5 events are subsequently signaled (4 · 20 s). The estimation of the maximum delay assumes that the office intrusion triggers unknown events. If this is (temporarily) not the case, the anomaly detection is delayed or disabled. The thresholds $T_{\text{significance}}$ and $T_{\text{max\_age}}$ have been determined in simulations. Various thresholds have been evaluated. The used values have performed well.

## 9.3 Anomaly Detection Performance

The last part of this chapter addresses the current deployment and its evaluation. As presented before, 10 sensing nodes have been deployed in an office offering two workplaces.

### 9.3.1 Office Occupancy Patterns

All experiments lasted between 2 and 4 days. Within these monitoring periods either normal office occupation or normal office occupation extended with specific hourly office access patterns were monitored. The specific patterns were either office searching performed by one person in an empty office room or hourly stress situations where multiple office staff were present, asking each other to look for missing items. The office access and occupation patterns are defined as follows:

- **Office searching**: The office is hourly searched for 2 - 5 minutes. The searching person enters the room and arbitrarily searches all different cabinets and drawers in the office. To avoid systematic search patterns, the office searching is performed by different persons. This pattern represents illegal access or abnormal behavior and is assumed to trigger alarms.
- **Stress situation**: In this office occupation pattern two to three persons are present in the office and are looking for some missing item(s). For example a document might be requested by an entering person and the two office personnel search and provide the requested information. Events of this kind last between 90 seconds and 3 minutes. This pattern imposes a high stress level on the system, but should not trigger alarms.
- **Normal office occupation**: Here, no restrictions have been defined. The office was just monitored for a given amount of time.

As discussed before, the system processes signals (PIR and vibration) in 20 s intervals on node level. Thus, every 20 s, local and global decisions are determined.

If nothing happens in the office or the activation is locally recognized, no event reports are sent to the base station. Hence, the base station has input vectors of form [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]. Such input vectors do not produce events at the base station. On the other hand, if many events occur during a certain time interval (office searching), many different unknown-event reports are sent to the base station. The resulting varying input vectors $\mathbf{z}$ cannot all be known at the base station. Hence, the base station signals accumulations of events which leads to the system-wide alarm (anomaly).

### 9.3.2 Computation at Desktop PC

The system based on Fuzzy ART neural networks at the sensor nodes and on a binary ART neural network at the base station has been implemented and evaluated in the sensor network. On the other hand, determining the set of applicable thresholds for the threshold-based local decider requires repeatable experiments. This cannot be provided by real-world experiments. Therefore, in addition to the running Fuzzy ART based system, every output of the sensor nodes was collected at a desktop PC, where the threshold-based anomaly detection system was implemented. Thus, the threshold-based decider could be evaluated and optimized offline with varying thresholds. To justify this implementation, the Fuzzy ART neural network based system was re-implemented at the base station, too. The results computed at the base station and at the desktop PC were equal. Hence, an offline analysis of the threshold-based decider is justified.

### 9.3.3 Detection Performance of the Anomaly Detectors

We investigated the number of false positives (false alarms) and false negatives (missing alarms) generated by the local Fuzzy ART and threshold-based (T-based) anomaly detectors. The threshold-based local anomaly detector has been evaluated with the complete range of applicable thresholds. Reports are generated if $\sum_{i=0}^{10} x_i > T$. Since analog input time vectors $\mathbf{x}$ are processed and single sensor readings are in a range from 0 to 50, the resulting threshold T could be in the range from 0 to 500. High values are improbable because too many signals would be filtered. Comparatively low values of 16 and 17 for T have shown good prevention of false alarms, while no real alarms have been missed. All three office occupation scenarios were evaluated: normal office occupation, hourly office searching and hourly stress situations. The first experiment lasted for 48 hours including two working days. The two other kinds of experiments lasted for 4 days. In these 4 days, the respective occupation pattern (searching | stress) was performed 8 times in 8 hours. Accordingly, the latter two experiments provided 32 specific office occupation patterns (searching | stress) in each case.

Table 9.1 shows the anomaly detection performance of both anomaly detectors. The first important result is that, apart from one failure of the threshold-based system with $T = 16$, no false negatives were observed in all experiments. This

**Table 9.1:** False Positives (FP) and False Negatives (FN) of the anomaly detectors.

| | Fuzzy ART | | T-based 16 | | T-based 17 | |
|---|---|---|---|---|---|---|
| | FP | FN | FP | FN | FP | FN |
| Normal office occupation (48h) | 1 | - | 1 | - | 1 | - |
| 32 hourly office searchings (96h) | - | - | 2 | 1 | 2 | - |
| 32 hourly stress situations (96h) | 1 | - | 12 | - | 9 | - |
| **Total** (240h) | **2** | - | **15** | **1** | **12** | - |

system behavior is very important, because false negatives mean undetected intrusions. The presence of false negatives would question any anomaly detection and alarming system. On the other hand, some false positives, i.e., false alarms, could not be prevented. In particular the hourly stress level experiments generated false alarms, whereby the threshold-based system performed much worse. If no false positives can be tolerated, the system could be used to trigger a secondary high-resolution system (see also Section 3.7). In such an implementation the presence of false positives is less severe since only the secondary system is unnecessarily triggered.

The reporting of false alarms in the hourly stress level experiments is due to similarity of these experiments and office searching. The experiments have shown that Fuzzy ART neural networks are able to recognize and filter local anomalies, which leads to the system-wide prevention of false alarms. We conclude this section by highlighting that the local event recognition feature of the Fuzzy ART neural networks is in particular beneficial in presence of high stress level without intrusion. In this case, the system based on Fuzzy ART neural networks led to 1 false alarm in 32 stress situations, whereas the system based on local thresholds reported between 9 and 12 false alarms in the same situations. No false negatives (missing alarms) were encountered by both systems.

### 9.3.4 Message Load

Only normal office occupation was evaluated to assess communication costs. The other two experiments do not reflect normal behavior. In these intrusion or stress situations artificial anomalies are generated, which leads to temporarily increased communication load compared to normal daily office occupation. Signaling every local input vector $\mathbf{x}$ with $\sum_{i=0}^{10} x_i > 0$ as event, i.e., threshold-based local decisions with T = 0, determines maximum possible communication costs. The costs of the respective anomaly detector have been computed in percentage of the maximum possible communication costs.

The communication costs of the Fuzzy ART system and the T-based system are

**Table 9.2:** Message load of the anomaly detectors.

| | Fuzzy ART | T-based 16 | T-based 17 |
|---|---|---|---|
| Monitoring without intrusion (48h) | 11.9 % | 7.0 % | 5.9 % |

shown in percentage in Table 9.2. The local filtering of the Fuzzy ART neural networks leads to communication cost savings of up to 90%. Even better results can be achieved with the threshold-based filtering . However, the T-based system requires learning and training, i.e., the thresholds need to be determined. In contrast, Fuzzy ART neural networks are completely self-learning, i.e., only the memory size and the vigilance factor $\rho$ need to be defined in advance. Furthermore, results in the last section have shown that the Fuzzy ART neural network approach prevents false alarms in case of stress situations.

Overall, without local filtering 10261 messages were reported to the base station during 48 hours. With the Fuzzy ART system this message load could be decreased to 1222 messages, which leads in average to 25 messages per hour. Considering the network size of 10 nodes and the faced office monitoring problem, a system-wide communication load of 25 messages per hour seems adequate.

### 9.3.5 Reporting and Triggering Delay

The reporting and triggering latency introduced by our anomaly detection system is investigated in this section. Section 9.2.2 has shown that the current implementation introduces a minimum reporting delay of 80 s and a maximum reporting delay of 320 s.

**Table 9.3:** Reporting Latencies [s] until alarm is reported.

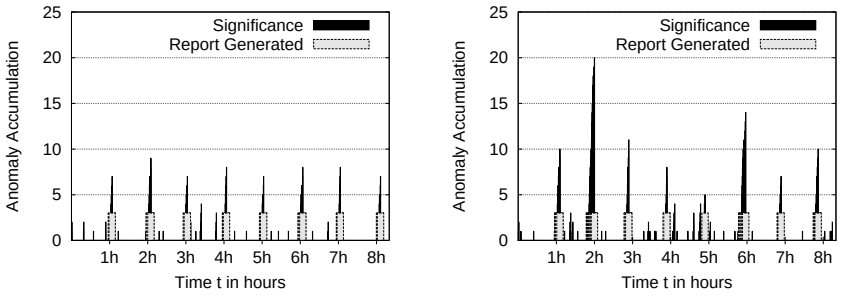
| | Fuzzy ART | | T-based 16 | | T-based 17 | |
|---|---|---|---|---|---|---|
| | $\mu$ | $\rho$ | $\mu$ | $\rho$ | $\mu$ | $\rho$ |
| Hourly office searching (32h) | 148 | 44 | 146 | 38 | 150 | 52 |

The effectively measured average reporting delays $\mu$ and their standard deviations $\rho$ are listed in Table 9.3. There is no significant difference between the different implementations. In average approximately 150 s were needed to detect and report office searching. The standard deviations vary slightly between 40 and 50 s. Detection latencies longer than 2 minutes seem rather long. However, this value depends on the duration of the monitoring period. The application of shorter monitoring periods could be evaluated. The monitoring period cannot be reduced arbitrarily due to the involved communication increase, though. The minimal re-

porting delay achievable is further restricted by the minimum amount of time required to identify abnormal behavior. We assume that reporting latencies of around 90 seconds might be achievable.

### 9.3.6 Standalone ART Neural Network Performance

In the previous experiments any local output has been reported to the base station to support later comparisons, i.e., to make the offline simulations of the threshold-based mechanism possible. On the other hand, a real deployment would only signal effectively unknown event patterns to the base station. For the sake of completeness, the Fuzzy ART based monitoring system has been run standalone.




**(a)** Soft hourly office searching.

**(b)** Intense hourly office searching.

**Figure 9.3:** Anomaly Detection Performance with ART neural networks.

Two monitoring periods of 8 hours are illustrated in Figure 9.3. The hourly office searching pattern has been used. In the first experiment the office was softly searched by making little noise, i.e., the office cabinets were carefully opened, searched and closed again. In the second experiment the office searching was more intensive. Little care was spent on avoiding noise. Accordingly, the sensor activation amplitudes were higher, which is confirmed in Figure 9.3. The intense office searching experiment signaled more reports of suspicious behavior. In both experiments all hourly office searching have been detected and no false alarms have been generated. The standalone implementation has shown similar performance to the implementation that has logged additional information.

## 9.4 Conclusions

A wireless sensor network for building monitoring has been proposed In this chapter. The system detects and reports abnormal office occupation. In contrast to conventional video surveillance systems, the system conceals the identity of the monitored office staff. Moreover, the deployed system is efficient and lightweight and produces much less data, decreasing administration and storage cost.

The system implements either a Fuzzy ART neural network or a simple threshold-based decider on a local level. A binary ART neural network is implemented at the base station. The proposed system with local Fuzzy ART decisions worked well in all experiments. In particular no false negatives were encountered, i.e., no cases of office searching were missed by the system. In normal office occupation with low stress level the threshold-based approach has performed similar to the Fuzzy ART neural network. Considering high stress levels, the Fuzzy ART neural network produces considerably fewer false positives than the threshold-based approach, though. This is due to the local memory maintained by the Fuzzy ART neural network that provides local recognition and filtering. It has been shown that communication costs could be cut by 90% with the Fuzzy ART-based system. The detection delay is currently approximately 2 minutes and 30 s, but could be further decreased by optimizing the monitoring cycles.

Optionally, the proposed system could be used to trigger a secondary, more detailed system such as a video surveillance system. Advantages of such an integrated approach would reduction of communication costs, post processing and storage, because the secondary, more expensive system would be only sporadically used. On the other hand, additional information could be captured on demand.

Chapter 10

# Conclusions

In this thesis the development of an energy-efficient, fault-tolerant and fully distributed event detection system has been addressed. The system needs to meet application-specific detection accuracy requirements. Nonetheless, energy-efficient mechanisms are required to support long-term operation. To optimize the trade-off between detection accuracy and long-term operation, appropriate functionality on different layers of the network stack is required. A smart integration of the functionality on the different layers is furthermore important. Therefore, we have suggested an architecture that implements a networking stack designed to satisfy the different system requirements. The MAC layer provides energy-efficient medium access control. In addition, synchronization messages that are periodically exchanged on the MAC layer have been used to discover neighborhood information and setup a routing backbone based on this information. By integrating routing functionality into the MAC layer, no additional control traffic to setup and maintain routing is required. Moreover, additional energy can be saved by temporarily turning off the radios of non-backbone nodes. Upon detection of an event, non-backbone nodes wake up and provide networking functionality to the overlay application software. Thus, medium access and routing are completely transparent to the application layer. Mobile networks are supported on the networking layer with an optional topology-adaptive routing mechanism. On top of these medium access and routing services the application-specific functionality has been implemented. Tracking groups are dynamically established and maintained upon detection of an event. The tracking groups provide the event localization and classification software executed at the leader node with the required information to perform these tasks. The classification software provides event classification and anomaly detection functionality. The event detection system has been applied to an office monitoring system that determines illegal or suspicious office access and occupancy. The individual chapters are summarized in the following.

In Chapter 4 we investigated the virtual clustering effect known from synchronized contention-based MAC protocols for wireless sensor networks. In virtual clustering, network nodes arrange themselves into groups according to common listen/sleep cycles. We have proposed a simple clock synchronization scheme that

avoids virtual clustering without introducing additional control traffic. All network nodes locally share a common listen/sleep schedule. The algorithm has shown fast and robust convergence. After convergence, no virtual clusters have appeared. It has been shown that the mechanism extends network lifetime and lowers storage complexity on the sensor nodes, because no tables with neighboring virtual clusters have to be stored.

In Chapter 5 the support of a routing backbone and the temporal shut-down of non-backbone nodes were focused on. We have provided two backbone construction methods (MPR-based CDS and N-CDS) that explore the content of the synchronization messages exchanged on the MAC layer to establish a CDS. By exploring the synchronization messages, no additional control traffic is generated. Both approaches have shown good energy load balancing and have performed well in reducing the current number of active nodes. Consequently, network lifetime is extended with both approaches. Network connectivity has been achieved in all simulations. In addition to the solutions on the MAC layer, we have implemented a CDS-based backbone construction method on the network layer (R-CDS). R-CDS requires the exchange of hello messages to learn local neighborhood. In terms of backbone construction, R-CDS performs similar to the solutions on the MAC layer. The advantage of R-CDS is the supply of backbone repair mechanisms. Thus, node mobility can be supported. On the other hand, R-CDS requires additional control traffic, which is waste of energy. Depending on network dynamics one of the approaches can be chosen.

In Chapter 6 the event detection and tracking functionality of DELTA has been provided. DELTA detects and tracks events efficiently. Considering networking, DELTA minimizes the number of event tracking groups. DELTA has been compared to EnviroTrack. EnviroTrack is an efficient and lightweight event detection and tracking mechanism. Both algorithms provide a similar set of basic operations. DELTA has outperformed EnviroTrack in detection speed and in avoiding concurrent tracking groups. DELTA has been able to detect and track events with varying sensing ranges efficiently with a single tracking group. Unlike EnviroTrack, DELTA provides the group leader with information needed to perform localization and classification. The message overhead of this data collection functionality has been evaluated in the real-world experiments and has shown to be the least possible. The data collection functionality further supports larger sensing ranges.

In Chapter 7 the localization and signal strength estimation performance of DELTA has been discussed. Based on event-relevant information collected in the tracking groups, position and emitted signal strengths of events have been estimated. The nonlinear Simplex Downhill (SD) method has shown to provide best performance considering the given requirements. In contrast to Linearized Least Square (LLS) methods, which require intrinsically less computation power, the SD method has proven to work also with the minimum amount of information needed to solve the resulting optimization problem. On the other hand, the computational burden has shown to be acceptable. In wireless sensor networks it might frequently happen that only the minimum amount of information about an event can be col-

lected due to packet loss or energy-efficiency reasons. These cases can only be covered with nonlinear methods, in our case with the SD method. The SD method achieved good estimation accuracy both in positioning and emitted signal strength estimations. Unlike the position estimations, the signal strength estimations are event specific and can be used for event classification.

In Chapter 8 the event classification functionality has been presented. All classifier software is performed at the DELTA leader node. The software supports event classification, i.e., the labeling of unknown events, and anomaly detection. In a first part the event classification software has been presented. This software addresses the classification of discrete event types. In experiments different kinds of light bulbs have been classified. Our own developed Fuzzy Logic Controller (FLC) has been compared to a simple Bayesian classifier and a Feedforward Neural Network (FFNN) approach. The FLC classifier has shown to be lightweight and accurate. Furthermore, the filtering of false alarms could be prevented with arbitrary accuracy. However, the more accurate the system is, the longer are the reporting delays due to filtering of events that do not satisfy the confidence threshold. Error rates of 5% could be achieved with low reporting latency. This is an accuracy improvement of 50% compared to the Bayesian classifier. Mainly due to storage complexity the presented classifiers are not appropriate for anomaly detection. For anomaly detection the application of ART neural networks has been proposed. ART neural networks represent a simple adaptive memory that is able to store and refine a certain number of prototypes. With ART neural networks it has been possible to compress and classify time series of event observations on sensor node level. Thus, communication costs can be kept low. The ART neural networks have shown to be very lightweight and sufficiently accurate. The ART neural networks have been used for office monitoring.

Finally, the office monitoring application has been presented in Chapter 9. Anomalies, locally determined by Fuzzy ART neural networks have been reported to a fusion center (the DELTA leader node), where the system wide decision has been implemented as a binary ART neural network. The system was able to report abnormal behavior (hourly office searching), while normal office access and occupation did not trigger any alarms. The message load of the proposed ART-based anomaly detector has shown to be marginal. In a normal office monitoring period of 48 hours approximately 25 messages per hour were transmitted, which reflects communication cost savings of 90%. The approach has been compared to a threshold-based anomaly detector. Both approaches have shown similar performance in normal office monitoring and in detecting office searching. However, under higher stress levels, the ART-based anomaly detector has outperformed the threshold-based approach. This is due to the local filtering capability of ART neural networks. In conclusion, the ART-based anomaly detector has shown to work reliably by introducing low communication costs. The anomaly detector can be used standalone or it can trigger a secondary, high-resolution system.

The main conclusions of this thesis can be summarized as follows. Existing monitoring applications are application-specific and are laid out for short-term de-

ployments. In contrast, we have proposed a common architecture for event detection that addresses accuracy while conserving energy. Energy savings have been achieved on different layers of the network stack. Based on a lightweight communication and group organization protocol, events have shown to be traceable with acceptable overhead. Furthermore, the amount of data required for classifications and decision making has been collected efficiently. Finally, feasible classification methods have been proposed, which have been used in an office monitoring application and have shown good performance, i.e., the system requirements in terms of energy savings and monitoring accuracy have been met.

Chapter 11

# Future Work

In this chapter possible directions for future work are outlined. First, application-specific system evaluation aspects are discussed. Second, some specific, unsolved problems and possible solutions are presented. Finally, some new trends and future directions of research are proposed.

The office monitoring system has been deployed in a small-scale office. Moreover, a rather small number of sensor nodes has been used. In future work larger scenarios could be tested. The performance of the system in an open-plan office could be investigated. This would impose a higher range of varying event patterns and more fluctuations. We assume that the local memory implemented on the sensor nodes should contribute even more in such a scenario than in small-scale environments. Nevertheless, the performance of the system needs to be demonstrated. Furthermore, different implementations of ART neural networks could be evaluated. Also, the memory size of the ART neural network could be adjusted depending on the application. In an open-plan office with a higher number of tolerable event patterns, the memory of the ART neural networks would probably have to be increased. Depending on the sensor platform used, this would also require some adaptations in the ART design. Finally, other kinds of event patterns could be included. Currently, the monitoring application has been run based on passive infrared and vibration events. In addition, noise levels could be monitored or illumination conditions, and so on.

So far, the different features of our monitoring system have been evaluated rather isolated. While testing DELTA in simulations and real-world experiments, the classification functionality of the framework had not yet been implemented. On the other hand, the final office monitoring system has imposed little stress on the DELTA tracking subsystem, because all sensor nodes were located in a single room within communication range of each other. In future work, the whole system could be implemented and evaluated in a large deployment which covers for example a whole department. In such a scenario, all features of our monitoring system could be tested concurrently.

Considering topology control, all proposed CDS approaches could be evaluated in comparison to each other. Until now, a detailed real-world performance analy-

sis of the topology control mechanisms is missing. Only simple scenarios have been tested. This is due to the suboptimal performing ESB sensor platform in this context. Accordingly, the most appropriate topology control mechanism could be implemented on a sensor node platform such as the TmoteSky platform. We assume that on such a platform the mechanisms should perform quite well. Finally, the topology control mechanisms implemented on the MAC layer have shown to outperform T-MAC, but no comparisons to other protocols have been performed. In future work, our protocols could be compared to both topology control and recent MAC protocols.

The gravity-based local clock synchronization scheme (LACAS) has been implemented as part of a cross-layer approach. Moreover, it has been evaluated in simulations only. In the current office monitoring system it has not been included. Therefore, LACAS could be implemented on sensor hardware and extensively tested in isolation to prove its benefits. Because the gravity-based mechanism is supposed to be very robust, good performance benchmarks can be expected.

We have shortly outlined some directions for future research in the protocols proposed in this thesis. Many more optimizations and configurations could be evaluated. It is a general property of real-world experiments that they are very time-consuming and continuously reveal additional challenges. In particular large-scale real-world wireless sensor network experiments are difficult to be done in a manageable way. On the other hand, simulations allow large-scale evaluations, but they are often not able to consider key properties of wireless sensor networks. Accordingly, also in this thesis some features have been evaluated in real-world experiments, while for others simulations have been performed.

The development of an accurate long-term-operating event detection system has led to some insights and ideas for future directions of research. In many existing systems networking aspects and application-specific requirements are still considered exclusively. Considering event detection, few systems address both, networking issues and detection accuracy. We think that more integrated approaches are needed to provide useful sensor network solutions to end users. A lot of research is spent on developing smart communication for wireless sensor networks. On the other hand, a target application is often not provided. However, the consideration of application-specific requirements is essential to develop useful systems. Therefore, we think that in future emphasize will have to be given to applications. So far, privacy has gained little attention in event detection systems. On the other hand, wireless sensor networks can be implemented to hide the identity of monitored personnel. We think that this could be a driving factor for the commercialization of a wireless sensor network based surveillance system. The usage of wireless sensor systems to trigger a high resolution system might be interesting and useful too.

Chapter 12

# Acronyms

| | |
|---|---|
| **ART** | Adaptive Resonance Theory |
| **BS** | Base Station |
| **CDS** | Connected Dominating Set |
| **CG** | Conjugate Gradient |
| **CSP** | Collaborative Signal Processing |
| **DELTA** | Distributed Event Localization and Tracking Algorithm |
| **DS** | Dominating Set |
| **ESB** | Embedded Sensor Board |
| **FFNN** | Feedforward Neural Network |
| **FSM** | Finite State Machine |
| **GPS** | Global Positioning System |
| **HPF** | High Pass Filter |
| **IBL** | Instance Based Learning |
| **IDSQ** | Information Driven Sensor Querying |
| **IREP** | Information Response |
| **LACAS** | Local Adaptive Clock Assimilation Scheme |
| **LLS** | Linear Least Square |
| **LPF** | Low Pass Filter |
| **MAC** | Medium Access Control |
| **MANET** | Mobile Ad Hoc Network |

| | |
|---|---|
| **MCDS** | Minimum Connected Dominating Set |
| **MIS** | Maxmimum Independent Set |
| **MPR** | Multipoint Relay |
| **NAV** | Network Allocation Vector |
| **PAR** | Photosynthetically Active Radiation |
| **QoS** | Quality of Service |
| **RSSI** | Received Signal Strength Indicator |
| **RTS/CTS** | Ready To Send / Clear To Send |
| **SD** | Simplex Downhill |
| **STM** | Short Term Memory |
| **TCP** | Transport Control Protocol |
| **TSR** | Total Solar Radiation |
| **TDMA** | Time Division Multiple Access |
| **WSN** | Wireless Sensor Network |

# Bibliography


[1] T. Abdelzaher, B. Blum, D. Evans, J. George, S. George, L. Gu, T. He, C. Huang, P. Nagaraddi, S. Son, P. Sorokin, J. Stankovic, and A. Wood, "EnviroTrack: Towards an environmental computing paradigm for distributed sensor networks," in *Proc. of the 24th International Conference on Distributed Computing Systems (ICDCS '04)*, Tokyo, Japan, 2004.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[3] ——, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[4] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *Proc. of the 1st international conference on Embedded networked sensor systems (SenSys '03)*, Los Angeles, California, USA, 2003, pp. 150–161.

[5] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "SensorScope: Out-of-the-box environmental monitoring," in *Proc. of the 7th International Conference on Information Processing in Sensor Networks (IPSN '08)*, Washington, DC, USA, 2008, pp. 332–343.

[6] E. A. Basha, S. Ravela, and D. Rus, "Model-based monitoring for early warning flood detection," in *Proc. of the 6th ACM conference on Embedded network sensor systems (SenSys '08)*, Raleigh, NC, USA, 2008, pp. 295–308.

[7] A. Basharat, N. Catbas, and M. Shah, "A framework for intelligent sensor network with video camera for structural health monitoring of bridges," in *Proc. of the Third IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW '05)*, Washington, DC, USA, 2005, pp. 385–389.

[8] A. Y. Benbasat and J. A. Paradiso, "A framework for the automated generation of power-efficient classifiers for embedded sensor nodes," in *Proc.*

*of the 5th international conference on Embedded networked sensor systems (SenSys '07)*, Sydney, Australia, 2007, pp. 219–232.

[9] F. U. Berlin, "Computer Systems & Telematics Group," December 2008. [Online]. Available: http://cst.mi.fu-berlin.de/projects/ScatterWeb/hardware/esb/powersupply%.htm

[10] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proc. of the 1st ACM international workshop on Wireless sensor networks and applications (WSNA '02)*, Atlanta, Georgia, USA, 2002, pp. 22–31.

[11] R. R. Brooks, P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks," in *Proc. of the IEEE*, vol. 91, no. 8, August 2003, pp. 1163–1171.

[12] R. Brooks, C. Griffin, and D. Friedlander, "Self-organized distributed sensor network entity tracking," *The International Journal of High Performance Computing Applications*, vol. 16, no. 3, pp. 207–219, 2002.

[13] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble mac protocol for duty-cycled wireless sensor networks," in *Proc. of the 4th international conference on Embedded networked sensor systems (SenSys '06)*, Boulder, Colorado, USA, 2006, pp. 307–320.

[14] N. Bulusu, D. Estrin, L. Girod, and J. Heidemann, "Scalable coordination for wireless sensor networks: Self-configuring localization systems," in *In Proceedings of the Sixth International Symposium on Communication Theory and Applications (ISCTA '01)*, Ambleside, Lake District, UK, 2001, pp. 1–6.

[15] T. Burwick and F. Joublin, "Optimal algorithmic complexity of Fuzzy ART," *Neural Processing Letters*, vol. 7, no. 1, pp. 37–41, February 1998.

[16] J. Cai, D. Ee, B. Pham, P. Roe, and J. Zhang, "Sensor network for the monitoring of ecosystem: Bird species recognition," in *In Proc. of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP '07)*, Melbourne, Australia, 2007, pp. 293–298.

[17] G. Carpenter and S. Grossberg, *Adaptive Resonance Theory*, ser. Bradford Books, M. A. Arbib, Ed. MIT Press, 2002.

[18] A. Cerpa and D. Estrin, "ASCENT: Adaptive self-configuring sensor networks topologies," *IEEE Transaction on Mobile Computing*, vol. 3, no. 3, pp. 272 – 285, July 2004.

[19] C.-C. G. Chang, W. E. Snyder, and C. Wang, "Robust localization of multiple events in sensor networks," in *Proc. of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)*, vol. 1, June 2006, pp. 168–177.

[20] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *Wireless Networks*, vol. 8, no. 5, pp. 481–494, 2002.

[21] M. Chen, V. C. M. Leung, S. Mao, and Y. Yuan, "Directional geographical routing for real-time video communications in wireless sensor networks," *Computer Communications*, vol. 30, no. 3368-3383, p. 17, 2007.

[22] S. H. Chi and T. H. Cho, *Fuzzy Systems and Knowledge Discovery*. Springer LNCS, 2006, ch. Fuzzy Logic Anomaly Detection Scheme for Directed Diffusion Based Sensor Networks, pp. 725–734.

[23] M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," *International Journal of High Performance Computing Applications*, vol. 16, no. 3, pp. 293–313, 2002.

[24] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs," *Discrete Mathematics*, vol. 86, no. 1-3, pp. 165–177, December 1990.

[25] T. Clausen and P. Jacquet, "Optimized link state routing protocol," RFC 3626, October 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3626.txt

[26] P. Costa, M. Migliavacca, G. P. Picco, and G. Cugola, "Epidemic algorithms for reliable content-based publish-subscribe: An evaluation," in *Proc. of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, Tokyo, Japan, 2004, pp. 552–561.

[27] A. P. R. da Silva, M. H. T. Martins, B. P. S. Rocha, A. A. F. Loureiro, L. B. Ruiz, and H. C. Wong, "Decentralized intrusion detection in wireless sensor networks," in *Proc. of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks (Q2SWinet '05)*, Montreal, Quebec, Canada, 2005, pp. 16–23.

[28] D. Dasgupta and S. Forrest, "Novelty detection in time series data using ideas from immunology," in *In Proc. of The Fifth International Conference on Intelligent Systems (IS '96)*, Reno, Nevada, USA, 1996.

[29] B. N. Delaunay, "Sur la sphère vide," *Bulletin of Academy of Sciences of the USSR*, vol. 7, no. 6, pp. 793–800, 1934.

[30] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *Communications Magazine*, vol. 44, no. 4, pp. 115–121, 2006.

[31] M. F. Duarte and Y. H. Hu, "Vehicle classification in distributed sensor networks," *Journal of Parallel and Distributed Computing*, vol. 64, no. 7, pp. 826–838, 2004.

[32] N. Dziengel, G. Wittenburg, and J. Schiller, "Towards distributed event detection in wireless sensor networks," in *In Proc. of the 4th IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS '08)*, Santorini Island, Greece, June 2008.

[33] A. El-Hoiydi and J.-D. Decotignie, "WiseMAC: An ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks," in *Proc. of the 9th IEEE International Symposium on Computers and Communications (ISCC 2004)*, vol. 1, Alexandria, Egypt, June 2004, pp. 244–251.

[34] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '01)*, vol. 4, Salt Lake City, UT, USA, 2001, pp. 2033–2036.

[35] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the physical world with pervasive networks," *IEEE Pervasive Computing*, vol. 1, no. 1, pp. 59–69, 2002.

[36] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *Proc. of the 5th annual ACM/IEEE international conference on Mobile computing and networking (MobiCom '99)*, Seattle, Washington, USA, 1999, pp. 263–270.

[37] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys (CSUR)*, vol. 35, no. 2, pp. 114–131, 2003.

[38] A. Faradjian, J. Gehrke, and P. Bonnet, "GADT: A probability space ADT for representing and querying the physical world," in *Proc. of the 18th International Conference on Data Engineering (ICDE '02)*, Washington, DC, USA, 2002, pp. 201–212.

[39] M. Friedman and A. Kandel, *Introduction to pattern recognition: statistical, structural, neural, and fuzzy logic approaches*, M. Friedman, Ed. World Scientific, 1999.

[40] K. R. Gabriel and R. R. Sokal, "A new statistical approach to geographic variation analysis," *Systematic Zoology*, vol. 18, pp. 259–270, 1969.

[41] A. Galstyan, B. Krishnamachari, K. Lerman, and S. Pattem, "Distributed online localization in sensor networks using a moving target," in *Proc. Third international symposium on Information processing in sensor networks (IPSN)*. Berkeley, California, USA: ACM, 2004, pp. 61–70.

[42] L. Gu, D. Jia, P. Vicaire, T. Yan, L. Luo, A. Tirumala, Q. Cao, T. He, J. A. Stankovic, T. Abdelzaher, and B. H. Krogh, "Lightweight detection and classification for wireless sensor networks in realistic environments," in *Proc. of*

*the 3rd international conference on Embedded networked sensor systems (SenSys '05)*, San Diego, California, USA, November 2005, pp. 205 – 217.

[43] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Proc. of the third international symposium on Information processing in sensor networks (IPSN '04)*, Berkeley, California, USA, 2004, pp. 1–10.

[44] S. Guha, R. N. Murty, and E. G. Sirer, "Sextant: A unified node and event localization framework using non-convex constraints," in *Proc. 6th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*. Urbana-Champaign, Illinois, USA: ACM, 2005, pp. 205–216.

[45] A. Haar, "Zur Theorie der orthogonalen Funktionensysteme," *Mathematische Annalen*, vol. 69, pp. 331–371, 1910.

[46] J. Hall, M. Barbeau, and E. Kranakis, "Anomaly-based intrusion detection using mobility profiles of public transportation users," in *Proc of the IEEE International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob'2005)*, vol. 2, Montreal, Canada, 2005, pp. 17–24.

[47] Z. He and D. Wu, "Resource allocation and performance analysis of wireless video sensors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 5, pp. 590 – 599, May 2006.

[48] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *Proc. of the 5th annual ACM/IEEE international conference on Mobile computing and networking (MobiCom '99)*, Seattle, Washington, US, 1999, pp. 174–185.

[49] M. Heissenbüttel, T. Braun, T. Bernoulli, and M. Wälchli, "BLR: Beaconless routing algorithm for mobile ad-hoc networks," *Computer Communications Journal*, vol. 27, no. 11, pp. 1076–1086, July 2004.

[50] M. Heissenbüttel, T. Braun, M. Wälchli, and T. Bernoulli, "Optimized stateless broadcasting in wireless multi-hop networks," in *Proc. 25th IEEE International Conference on Computer Communications (INFOCOM)*, Barcelona, Spain, April 2006, pp. 1–12.

[51] M. Heissenbüttel, T. Braun, M. Wälchli, and T. Bernoulli, "Evaluating the limitations of and alternatives in beaconing," *Ad-Hoc Networks*, vol. 5, no. 5, pp. 558–578, 2007.

[52] J. Hightower and G. Borriella, "Location systems for ubiquitous computing," *IEEE Computer*, vol. 34, no. 8, pp. 57–66, 2001.

[53] G. Hoblos, M. Staroswiecki, and A. Aitouche, “Optimal design of fault tolerant sensor networks,” in *Proc. of the 2000 IEEE International Conference on Control Applications (CCA ’00)*, Anchorage, AK, USA, 2000, pp. 467–472.

[54] C. Hu, S. Xu, and X. Yang, “A review on interval computation - software and applications,” *International Journal of Computational and Numerical Analysis and Applications*, vol. 1, no. 2, pp. 149–162, 2002.

[55] T. Instruments, “Infrared light-to-frequency converter,” Mai 2007. [Online]. Available: http://focus.ti.com/lit/ds/soes018/soes018.pdf

[56] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed diffusion: a scalable and robust communication paradigm for sensor networks,” in *Proc. of the 6th annual international conference on Mobile computing and networking (MobiCom ’00)*, Boston, Massachusetts, United States, 2002, pp. 56–67.

[57] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, “Directed diffusion for wireless sensor networking,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 2–16–, 2003.

[58] X. Ji and H. Zha, “Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling,” in *Proc. of the twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM ’04)*, vol. 4, Hong Kong, China, March 2004, pp. 2652– 2661.

[59] D. B. Johnson and D. A. Maltz, *Dynamic source routing in ad hoc wireless networks*, S. US, Ed. Mobile Computing, 1996, vol. 353.

[60] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*, Wiley, Ed. John Wiley & Sons, 2005.

[61] B. Karp and H. T. Kung, “GPSR: greedy perimeter stateless routing for wireless networks,” in *Proc. of the 6th annual international conference on Mobile computing and networking (MobiCom ’00)*, Boston, Massachusetts, US, 2000, pp. 243–254.

[62] M. Khan, G. Pandurangan, and V. S. A. Kumar, “Distributed algorithms for constructing approximate minimum spanning trees in wireless sensor networks‘,” *IEEE Transactions on Parallel Distributed Systems*, vol. 20, no. 1, pp. 124–139, 2009.

[63] L. E. Kinsler, A. R. Frey, A. B. Coppens, and J. V. Sanders, *Fundamentals of Acoustics*, Wiley, Ed. Wiley and Sons, Inc., 2000.

[64] B. Krishnamachari and S. Iyengar, “Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks,” *IEEE Transactions on Computers*, vol. 53, no. 3, pp. 241–250, 2004.

[65] I. Krontiris, T. Dimitriou, T. Giannetsos, and M. Mpasoukos, *Algorithmic Aspects of Wireless Sensor Networks*. Springer Berlin / Heidelberg, 2008, ch. Intrusion Detection of Sinkhole Attacks in Wireless Sensor Networks, pp. 150–161.

[66] A. Kuh1, C. Zhu1, and D. Mandic, *Knowledge-Based Intelligent Information and Engineering Systems*. Springer LNCS, 2006, ch. Sensor Network Localization Using Least Squares Kernel Regression, pp. 1280–1287.

[67] A. Kulakov and D. Davcev, "Distributed data processing in wireless sensor networks based on artificial neural-networks algorithms," in *Proc. of the 10th IEEE Symposium on Computers and Communications (ISCC '05)*, La Manga del Mar Menor, Cartagena, Spain, 2005, pp. 353–358.

[68] ——, "Tracking of unusual events in wireless sensor networks based on artificial neural-networks algorithms," in *Proc. of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, Las Vegas, Nevada, USA, 2005, pp. 534–539.

[69] ——, "Intelligent wireless sensor networks using FuzzyART neural-networks," in *Proc. of the 12th IEEE Symposium on Computers and Communications (ISCC '07)*, Aveiro, Portugal, July 2007, pp. 569–574.

[70] P. Kulkarni, P. J. Shenoy, and D. Ganesan, "Approximate initialization of camera sensor networks," in *Proc. of the 4th European conference on Wireless Sensor Networks (EWSN 2007)*, Delft, The Netherlands, 2007, pp. 67–82.

[71] M. Kumar, L. Schwiebert, and M. Brockmeyer, "Efficient data aggregation middleware for wireless sensor networks," in *Proc. of the First International Conference on Mobile, Ad-Hoc, and Sensor Systems (MASS '04)*, Fort Lauderdale, Florida, USA, October 2004, pp. 579–581.

[72] L. I. Kuncheva, *Fuzzy Classifier Design*, ser. Studies in Fuzziness and Soft Computing, J. Kacprzyk, Ed. Physica-Verlag, A. Springer-Verlag Company, 2000, vol. 49.

[73] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitive comparison," *Computer Networks*, vol. 43, no. 4, pp. 499–518, 2003.

[74] D.-U. Lee, H. Kim, S. Tu, M. Rahimi, D. Estrin, and J. D. Villasenor, "Energy-optimized image communication on resource-constrained sensor platforms," in *Proc. of the 6th international conference on Information processing in sensor networks (IPSN '07)*, Cambridge, Massachusetts, USA, 2007, pp. 216–225.

[75] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *The Quarterly of Applied Mathematics*, vol. 2, pp. 164–168, 1944.

[76] D. Li and Y. H. Hu, "Energy-based collaborative source localization using acoustic microsensor array," *EURASIP Journal on Applied Signal Processing*, vol. 3, no. 1, pp. 321–337, 2003.

[77] ——, "Least square solutions of energy based acoustic source localization problems," in *Proc. of the International Conference on Parallel Processing Workshops (ICPPW '04)*, Washington, DC, USA, 2004, pp. 443–446.

[78] D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed, "Detection, classification and tracking of targets," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 17–29, March 2002.

[79] M. Li and Y. Liu, "Underground structure monitoring with wireless sensor networks," in *Proc. of the 6th international conference on Information processing in sensor networks (IPSN '07)*, Cambridge, Massachusetts, USA, 2007, pp. 69–78.

[80] N. Li, J. Hou, and L. Sha, "Design and analysis of an MST-based topology control algorithm," *IEEE Transactions on Wireless Communications*, vol. 4, no. 3, pp. 1195– 1206, 2005.

[81] S. Li, S. H. Son, and J. A. Stankovic, "Event detection services using data service middleware in distributed sensor networks," in *Proc. of the second international symposium on Information processing in sensor networks (IPSN '03)*, Palo Alto, USA, April 2003, pp. 502–517.

[82] X.-Y. Li, P.-J. Wan, Y. Wang, and O. Frieder, "Sparse power efficient topology for wireless networks," in *Proc. of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, Washington, DC, USA, 2002, pp. 3839 – 3848.

[83] X.-Y. Li, G. Calinescu, and P.-J. Wan, "Distributed construction of a planar spanner and routing for ad hoc wireless networks," in *Proc. of the twenty-first Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, vol. 3, no. 1268-1277, New York, US, 2002.

[84] X.-Y. Li, Y. Weng, P.-J. Wan, W.-Z. Song, and O. Frieder, "Localized low-weight graph and its applications in wireless ad hoc networks," in *Proc. of the twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 04)*, vol. 1, Hong Kong, China, 2004, pp. 431–442.

[85] Y. Li, W. Ye, and J. Heidemann, "Energy and latency control in low duty cycle mac protocols," in *Proc. of the IEEE Wireless Communications and*

*Networking Conference (WCNC '05)*, New Orleans, LA, March 2005, pp. 676–682.

[86] Y. Y. Li and L. Parker, "Intruder detection using a wireless sensor network with an intelligent mobile robot response," in *Proc. of the IEEE SoutheastCon 2008*, Huntsville, Alabama, USA, 2008, pp. 37–42.

[87] J. Liu, M. Chu, J. Liu, J. Reich, and F. Zhao, "Distributed state representation for tracking problems in sensor networks," in *Proc. of the third international symposium on Information processing in sensor networks (IPSN '04)*, Berkeley, California, USA, 2004, pp. 234–242.

[88] Y. Liu and L. M. Ni, "A new MAC protocol design for long-term applications in wireless sensor networks," in *Proc. of the 13th International Conference on Parallel and Distributed Systems (ICPADS '07)*, Hsinchu, Taiwan, 2007, pp. 1–8.

[89] W. Luh, D. Kundur, and T. Zourntos, "A novel distributed privacy paradigm for visual sensor networks based on sharing dynamical systems," *EURASIP Journal of Applied Signal Processing*, vol. 2007, no. 1, pp. 1–17, 2007.

[90] L. Luo, T. F. Abdelzaher, T. He, and J. A. Stankovic, "EnviroSuite: An environmentally immersive programming framework for sensor networks," *ACM Transaction on Embedded Computing System (TECS)*, vol. 5, no. 3, pp. 543–576, 2006.

[91] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: a tiny aggregation service for ad-hoc sensor networks," in *OSDI*, Boston, USA, December 2002.

[92] ——, "The design of an acquisitional query processor for sensor networks," in *Proc. of the 2003 ACM SIGMOD international conference on Management of data (SIGMOD '03)*, San Diego, California, June 2003, pp. 491–502.

[93] ——, "TinyDB: An acquisitional query processing system for sensor networks," *ACM Transactions on Database Systems*, vol. 30, no. 1, pp. 122–173, 2005.

[94] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. of the 1st ACM international workshop on Wireless sensor networks and applications (WSNA '02)*, Atlanta, Georgia, USA, 2002, pp. 88–97.

[95] E. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975.

[96] N. Mazhar and M. Farooq, *Artificial Immune Systems*. Springer LNCS, 2007, ch. BeeAIS: Artificial Immune System Security for Nature Inspired, MANET Routing Protocol, BeeAdHoc, pp. 370–381.

[97] ——, "A sense of danger: dendritic cells inspired artificial immune system for MANET security," in *Proc. of the 10th annual conference on Genetic and evolutionary computation (GECCO '08)*, Atlanta, GA, USA, 2008, pp. 63–70.

[98] M. Meer, "The delta object tracking and localization algorithm for sensor networks," Master's thesis, University of Bern, 2006.

[99] C. Meesookho, S. Narayanan, and C. Raghavendra, "Collaborative classification applications in sensor networks," in *Proc. of the Sensor Array and Multichannel Signal Processing Workshop (SAM '02)*, Rosslyn, VA, USA, 2002, pp. 370–374.

[100] Mobility-fw, "Mobility framework for omnet++," October 2006. [Online]. Available: http://mobility-fw.sourceforge.net/

[101] MPI, "Message passing interface forum." December 2008. [Online]. Available: http://www.mpi-forum.org/index.html

[102] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar, "Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the COMPOW protocol," in *Proc. of the European Wireless Conference (EW '02)*, Florence, Italy, 2002, pp. 156–162.

[103] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, no. 1, pp. 308–313, 1965.

[104] X. Nguyen, M. I. Jordan, and B. Sinopoli, "A kernel-based learning approach to ad hoc sensor network localization," *ACM Transactions on Senor Networks*, vol. 1, no. 1, pp. 134–152, 2005.

[105] D. Niculescu, "Positioning in ad hoc sensor networks," *IEEE Communications Magazine*, vol. 18, no. 4, pp. 24–29, July/August 2004.

[106] D. Niculescu and B. Nath, "Ad hoc positioning system (APS)," in *Proc. of the Global Telecommunications Conference (GLOBECOM '01)*, San Antonio, USA, 2001, pp. 2926 – 2931.

[107] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[108] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. of the Second IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '99)*, New Orleans, LA, USA, 1999, pp. 90–100.

[109] C. E. Perkins, *Ad hoc networking*, C. E. Perkins, Ed. Addison-Wesley Longman Publishing Co., Inc., 2008.

[110] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," *SIGCOMM Computer Communication Review*, vol. 24, no. 4, pp. 234–244, 1994.

[111] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*, Baltimore, MD, USA, November 2004, pp. 95–107.

[112] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[113] G. Powell, D. Marshall, P. Smets, B. Ristic, and S. Maskell, "Joint tracking and classification of airbourne objects using particle filters and the continuous transferable belief model," in *Proc. of the 9th International Conference on Information Fusion (Fusion '06)*, Florence, Italy, July 2006, pp. 1–8.

[114] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, W. H. Press and S. A. Teukolsky, Eds. Cambridge University Press, 1992.

[115] N. B. Priyantha, A. K. Miu, H. Balakrishnan, and S. Teller, "The cricket compass for context-aware mobile applications," in *Proc. of the 7th annual international conference on Mobile computing and networking (MobiCom '01)*, Rome, Italy, 2001, pp. 1–14.

[116] A. Quayyum, L. Viennot, and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks," INRIA, Sophia Antipolis, France, Tech. Rep., 2000.

[117] J. Rabaey, J. Ammer, J. L. da Silva Jr., and D. Patel, "PicoRadio: Ad-hoc wireless networking of ubiquitous low-energy sensor/monitor nodes," in *Proc. of the IEEE Computer Society Annual Workshop on VLSI (WVLSI '00)*, Orlando, FL, USA, 2000, pp. 9–12.

[118] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. of the third international symposium on Information processing in sensor networks (IPSN '04)*, Berkeley, California, USA, 2004, pp. 20–27.

[119] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves, "Energy-efficient collision-free medium access control for wireless sensor networks," in *Proc. of the 1st international conference on Embedded networked sensor systems (SenSys '03)*, Los Angeles, CA, 2003, pp. 181 – 192.

[120] P. Ramanathan, K. C. Wang, K. K. Saluja, and T. Clouqueur, "Communication support for location-centric collaborative signal processing in sensor networks," in *Proc. of DIMACS Workshop on Pervasive Networks*, 2001.

[121] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with GHT, a geographic hash table," *Mobile Networks and Applications*, vol. 8, no. 4, pp. 427–442, 2003.

[122] M. Roberts and D. Marshall, "Using classification to improve wireless sensor network management with the continuous transferable belief model," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 7112, 2008.

[123] R. Roman, J. Zhou, and J. Lopez, "Applying intrusion detection systems to wireless sensor networks," in *Proc. of the 3rd IEEE Consumer Communications and Networking Conference (CCNC '06)*, vol. 1, Las Vegas, Nevada, USA, January 2006, pp. 640–644.

[124] K. Römer, "Discovery of frequent distributed event patterns in sensor networks," in *Proc. of the 5th European Conference on Wireless Sensor Networks (EWSN '08)*, Bologna, Italy, 2008, pp. 106–124.

[125] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proc. of the 7th annual international conference on Mobile computing and networking (MobiCom '01)*, Rome, Italy, 2001, pp. 166–179.

[126] A. Savvides, H. Park, and M. B. Srivastava, "The n-hop multilateration primitive for node localization problems," *Mobile Networks and Applications*, vol. 8, no. 4, pp. 443–451, 2003.

[127] Scatterweb, "The self-organizing wireless communication platform," October 2007. [Online]. Available: http://www.scatterweb.net

[128] Sentilla, "Pervasive computing solutions," http://www.sentilla.com/, July 2008. [Online]. Available: http://www.sentilla.com/moteiv-transition.html

[129] Y. Shang and W. Rhum, "Improved MDS-based localization," in *Proc. of the twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 4, Hong Kong, China, March 2004, pp. 2640– 2651.

[130] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from mere connectivity," in *Proc. of the 4th ACM international symposium on Mobile ad hoc networking & computing (MobiHoc '03)*, Annapolis, Maryland, USA, 2003, pp. 201–212.

[131] X. Sheng and Y.-H. Hu, "Energy based acoustic source localization," in *Proc. of the 2nd International Workshop on Information Processing in Sensor Networks (IPSN '03)*, Palo Alto, CA, USA 2003, pp. 285–300.

[132] ——, "Maximum likelihood multiple-source localization using acoustic energy measurements with wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 53, no. 1, pp. 44– 53, January 2005.

[133] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, "Wireless adhoc sensor and actuator networks on the farm," in *Proc. of the 5th international conference on Information processing in sensor networks (IPSN '06)*, Nashville, Tennessee, USA, 2006, pp. 492–499.

[134] G. Simon, G. Balogh, G. Bap, M. Maróti, B. Kusy, J. Sallai, A. Lédeczi, A. Nádas, and K. Frampton, "Sensor network-based countersniper system," in *Proc. of the 2nd international conference on Embedded networked sensor systems (SenSys '04)*, Baltimore, Maryland, USA, November 2004, pp. 1–12.

[135] J. O. Smith and J. S. Abel, "Closed-form least-squares source location estimation from range-difference measurements," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 35, no. 12, pp. 1661– 1669, December 1987.

[136] J. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou, "Real-time communication and coordination in embedded sensor networks," *Proceedings of the IEEE*, vol. 91, no. 7, pp. 1002–1022, 2003.

[137] T. Staub, T. Bernoulli, M. Anwander, M. Wälchli, and T. Braun, "Experimental lifetime evaluation for MAC protocols on real sensor hardware," in *Proc. of ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'06)*, Uppsala, Sweden, 2006.

[138] C. Suh and Y.-B. Ko, "A traffic aware, energy efficient MAC protocol for wireless sensor networks," in *Proc of the IEEE 2005 International Symposium on Circuits and Systems (ISCAS'05)*, vol. 3, Kobe, Japan, May 2005, pp. 2975– 2978.

[139] Y. Sun, S. Du, O. Gurewitz, and D. B. Johnson, "DW-MAC: a low latency, energy efficient demand-wakeup MAC protocol for wireless sensor networks," in *Proc. of the 9th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '08)*, Hong Kong, Hong Kong, China, 2008, pp. 53–62.

[140] Y. Sun, O. Gurewitz, and D. B. Johnson, "RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proc. of the 6th ACM conference on Embedded network sensor systems (SenSys '08)*, Raleigh, NC, USA, 2008, pp. 1–14.

[141] G. T. Toussaint, "The relative neighborhood graph of a finite planar set," *Pattern Recognition*, vol. 12, pp. 261–268, 1980.

[142] T. van Dam and K. Langendoen, "An adaptive energy-efficient mac protocol for wireless sensor networks," in *Proc. of the 1st international conference on Embedded networked sensor systems (SenSys '03)*, Los Angeles, CA, 2003, pp. 171–180.

[143] L. van Hoesel and P. Havinga, "A lightweight medium access protocol (LMAC) for wireless sensor networks: Reducing preamble transmissions and transceiver state switches," in *Proc. of the International Conference on Networked Sensing Systems (INSS '04)*, Tokyo, Japan, June 2004.

[144] A. Varga, "OMNeT++: a component-based, modular and open-architecture simulation environment," December 2008. [Online]. Available: http://www.omnetpp.org/

[145] S. Venkatesh and R. M. Buehrer, "A linear programming approach to NLOS error mitigation in sensor networks," in *Proc. of the fifth international conference on Information processing in sensor networks (IPSN '06)*, Nashville, Tennessee, USA, 2006, pp. 301–308.

[146] P. Volgyesi, G. Balogh, A. Nadas, C. B. Nash, and A. Ledeczi, "Shooter localization and weapon classification with soldier-wearable networked sensors," in *Proc. of the 5th international conference on Mobile systems, applications and services (MobiSys '07)*, San Juan, Puerto Rico, 2007, pp. 113–126.

[147] M. Wälchli, "Optimized position-based routing and broadcasting in mobile ad-hoc networks," Master's thesis, University of Bern, 2004.

[148] M. Wälchli, T. Bernoulli, and T. Braun:, "Receiver-based backbone construction and maintenance for wireless sensor or multi-hop networks," in *Proc. of the Workshop on Mobile Ad-Hoc Networks (WMAN 2007)*, Bern, Switzerland, March 2007, pp. 409–420.

[149] M. Wälchli, S. Bissig, and T. Braun, "Intensity-based object localization and tracking with wireless sensors," in *Proc. of the ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'06)*, Uppsala, Sweden, June 2006.

[150] M. Wälchli, S. Bissig, M. Meer, and T. Braun, "Distributed event tracking and classification in wireless sensor networks," *Journal of Internet Engineering*, vol. 2, no. 1, pp. 117–126, 2008.

[151] M. Wälchli and T. Braun, "Receiver-based CDS algorithm for wireless networks," in *Proc. of the 5. GI/ITG KuVS Fachgespräche Drahtlose Sensornetze '06*, Stuttgart, Germany, 2006, pp. 69–73.

[152] ——, "Event classification and filtering of false alarms in wireless sensor networks," in *In Proc. of 3rd International Workshop on Intelligent Systems Techniques for Ad hoc and Wireless Sensor Networks (IST-AWSN'08)*, Sydney, Australia, December 2008.

[153] ——, "Efficient signal processing and anomaly detection in wireless sensor networks," in *Proc. of 6th European Workshop on Nature-inspired Techniques for Telecommunications Networks and other Parallel and Distributed Systems (EvoComent '09)*, Tübingen, Germany, April 2009.

[154] ——, "Office access monitoring with a wireless sensor network," in *submitted to the 1st ICST AdHocNets 2009*, 2009.

[155] M. Wälchli, M. Scheidegger, and T. Braun, "Intensity-based event localization in wireless sensor networks," in *Proc. of the Third IFIP Annual Conference on Wireless On Demand Network Systems and Services (WONS '06)*, Les ménuires, France, 2006, pp. 41–49.

[156] M. Wälchli, P. Skoczylas, M. Meer, and T. Braun, "Distributed event localization and tracking with wireless sensors," in *Proc. of the 5th International Conference on Wired/Wireless Internet Communications (WWIC '07)*, Coimbra, Portugal, May 2007, pp. 247–258.

[157] M. Wälchli, R. Zurbuchen, T. Staub, and T. Braun, "Coordinated sleep MAC for energy-constrained wireless sensor networks," in *submitted to the 34th IEEE LCN 2009*, 2009.

[158] ——, "Gravity-based local clock synchronization in wireless sensor networks," in *Proc. of the 8th International Conference on Networking (Networking 2009)*, Aachen, Germany, May 2009.

[159] P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," *Mobile Networks and Applications*, vol. 9, no. 2, pp. 141–149, 2004.

[160] T.-Y. Wang, Y. S. Han, P. K. Varshney, and P.-N. Chen, "Distributed fault-tolerant classification in wireless sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 724–734, 2005.

[161] X. R. Wang, J. T. Lizier, O. Obst, M. Prokopenko, and P. Wang, "Spatiotemporal anomaly detection in gas monitoring sensor networks," in *Proc. of the 5th European Conference on Wireless Sensor Networks (EWSN '08)*, Bologna, Italy, 2008, pp. 90–105.

[162] G. Werner-Allen, S. Dawson-Haggerty, and M. Welsh, "Lance: optimizing high-resolution signal collection in wireless sensor networks," in *Proc. of the 6th ACM conference on Embedded network sensor systems (SenSys '08)*, Raleigh, NC, USA, 2008, pp. 169–182.

[163] J. Williams, J. Fisher, and A. Willsky, "Approximate dynamic programming for communication-constrained sensor network management," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4300–4311, 2007.

[164] G. Wittenburg, N. Dziengel, and J. Schiller, "In-network training and distributed event detection in wireless sensor networks," in *Proc. of the 6th ACM conference on Embedded network sensor systems (SenSys '08)*, Raleigh, NC, USA, 2008, pp. 387–388.

[165] G. Wittenburg, K. Terfloth, F. L. Villafuerte, T. Naumowicz, H. Ritter, and J. Schiller, "Fence monitoring - experimental evaluation of a use case for wireless sensor networks," in *Proc. of the Fourth European Conference on Wireless Sensor Networks (EWSN '07)*, Delft, The Netherlands, 2007, pp. 163–178.

[166] J. Wu, F. Dai, M. Gao, and I. Stojmenovic, "On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks," *IEEE/KICS Journal of Communications and Networks*, vol. 4, no. 1, pp. 59–70, March 2002.

[167] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," in *Proc. of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications (DIALM '99)*, Seattle, WA, 1999, pp. 7–14.

[168] D. Xie, T. Yan, D. Ganesan, and A. Hanson, "Design and implementation of a dual-camera wireless sensor network for object retrieval," in *Proc. of the 7th international conference on Information processing in sensor networks (IPSN '08)*, Washington, DC, USA, 2008, pp. 469–480.

[169] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad-hoc routing," in *Proc. of the 7th annual international conference on Mobile computing and networking (MobiCom '01)*, Rome, Italy, July 2001, pp. 70–84.

[170] X. Yang, H. B. Lim, T. M. Özsu, and K. L. Tan, "In-network execution of monitoring queries in sensor networks," in *Proc. of the 2007 ACM SIGMOD international conference on Management of data (SIGMOD '07)*, Beijing, China, 2007, pp. 521–532.

[171] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," *SIGMOD Records*, vol. 31, no. 3, pp. 9–18, 2002.

[172] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. of Proceedings 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, vol. 3, New York, US, 2002, pp. 1567–1576.

[173] ——, “Medium access control with coordinated adaptive sleeping for wireless sensor networks,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 493–506, June 2004.

[174] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, “Hardware design experiences in ZebraNet,” in *Proc. of the 2nd international conference on Embedded networked sensor systems (SenSys ’04)*, Baltimore, MD, USA, 2004, pp. 227–238.

[175] J. Zhao and R. Govindan, “Understanding packet delivery performance in dense wireless sensor networks.” in *Proc. of the First ACM Conference on Embedded Networked Sensor Systems (SenSys ’03)*, Los Angeles, CA, November 2003, pp. 1–13.

[176] T. Zheng, S. Radhakrishnan, and V. Sarangan, “PMAC: An adaptive energy-efficient mac protocol for wireless sensor networks,” in *Proc. of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS’05)*, 2005, pp. 95–107.

[177] D. Zhou, M.-T. Sun, and T.-H. Lai, “A timer-based protocol for connected dominating set construction in IEEE 802.11 multihop mobile ad hoc networks,” in *Proc. of the 2005 Symposium on Applications and the Internet (SAINT’05)*, 2005, pp. 2–8.

[178] C. Zhu and A. Kuh, “Ad hoc sensor network localization using distributed kernel regression algorithms,” in *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, vol. 2, Honolulu, Hawai, US, 2007, pp. 497–500.

[179] Y. Zou and K. Chakrabarty, “Sensor deployment and target localization in distributed sensor networks,” *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 1, pp. 61–91, February 2004.

[180] R. Zurbuchen, “Backbone construction based on sync-messages and energy-savings in wireless sensor networks,” Master’s thesis, University of Bern, 2009.

# Erklärung

gemäss Art. 28 Abs. 2 RSL 05

| | |
|---|---|
| Name/Vorname: | Markus Wälchli |
| Matrikelnummer: | 00-112-573 |
| Studiengang: | Informatik |
| | Bachelor ☐ Master ☐ Dissertation ☒ |
| Titel der Arbeit: | Distributed Event Detection in Wireless Sensor Networks |
| Leiter der Arbeit: | Prof. Dr. Torsten Braun |

Ich erkläre hiermit, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet. Mir ist bekannt, dass andernfalls der Senat gemäss Artikel 36 Absatz 1 Buchstabe o des Gesetzes vom 5. September 1996 über die Universität zum Entzug des auf Grund dieser Arbeit verliehenen Titels berechtigt ist.

Bern, 04.05.2009

.....

Unterschrift

# Curriculum Vitae

| | |
|---|---|
| 1976 | Born on September 26, in Bern, Switzerland |
| 1983-1992 | Primary school in Niederwangen, Bern |
| 1992-1993 | $10^{th}$ school year at BBF Bern, Bern |
| 1993-1997 | Apprenticeship as an electrician, Linder + Lötscher AG, Bern |
| | Professional high school (Berufsmatura), Bern, Switzerland |
| 1998-2000 | Berner Maturitätsschule für Erwachsene (BME), |
| | Bern-Neufeld, Typus C |
| 2000-2005 | University of Bern, Switzerland, Major in Computer Science |
| | and Minors in Mathematics and Philosophy |
| 2003 | MICS summer internship for undergraduate students, Bern |
| 2005-2009 | Research Assistant and Ph.D. Student at the |
| | Institute for Computer Science |
| | and Applied Mathematics, University of Bern |