

A DISTRIBUTED DATA EXTRACTION AND VISUALISATION SERVICE FOR WIRELESS SENSOR NETWORKS

MOHAMMAD HAMMOUDEH

A thesis submitted in partial fulfilment of the requirements of the
University of Wolverhampton for the degree of Doctor of Philosophy

2008

This work or any part thereof has not previously been presented in any form to the University or to any other body whether for the purposes of assessment, publication or for any other purpose (unless otherwise indicated). Save for any express acknowledgements, references and/or bibliographies cited in the work, I confirm that the intellectual content of the work is the result of my own efforts and of no other person.

The right of Mohammad Hammoudeh to be identified as author of this work is asserted in accordance with ss.77 and 78 of the Copyright, Designs and Patents Act 1988. At this date copyright is owned by the author.

Signature.....

Date.....

Abstract

With the increase in applications of wireless sensor networks, data extraction and visualisation have become a key issue to develop and operate these networks. Wireless sensor networks typically gather data at a discrete number of locations. By bestowing the ability to predict inter-node values upon the network, it is proposed that it will become possible to build applications that are unaware of the concrete reality of sparse data.

The aim of this thesis is to develop a service for maximising information return from large scale wireless sensor networks. This aim will be achieved through the development of a distributed information extraction and visualisation service called the *mapping service*. In the distributed mapping service, groups of network nodes cooperate to produce local maps which are cached and merged at a sink node, producing a map of the global network. Such a service would greatly simplify the production of higher-level information-rich representations suitable for informing other network services and the delivery of field information visualisations.

The proposed distributed mapping service utilises a blend of both inductive and deductive models to successfully map sense data and the universal physical principles. It utilises the special characteristics of the application domain to render visualisations in a map format that are a precise reflection of the concrete reality. This service is suitable for visualising an arbitrary number of sense modalities. It is capable of visualising from multiple independent types of the sense data to overcome the limitations of generating visualisations from a single type of a sense modality. Furthermore, the proposed mapping service responds to changes in the environmental conditions that may impact the visualisation performance by continuously updating the application domain model in a distributed manner. Finally, a new distributed self-adaptation algorithm, Virtual Congress Algorithm, which is based on the concept of virtual congress is proposed, with

the goal of saving more power and generating more accurate data visualisation.

Acknowledgements

I wish to express my greatest thanks to my director of studies, Prof. Robert Newman, for his insightful directions, thoughtful discussions and heartily support. I also want to thank the other members in my doctoral supervision team. I would especially like to thank Miss. Sarah Mount and Dr. Christopher Dennett, there supervision has been invaluable and my life has been enriched personally and professionally by working with them.

Special thanks to Dr. Alexander Kurz, Dr. James Shuttleworth, and Dr. Emilio Tuosto.

My thanks are due to my mother for her unlimited love and support. She taught me the value of knowledge and has stood by me at all times, providing constant support, encouragement and love. Finally, I would like to thank my wonderful wife Hanan for her patience and smile in the face of weekends I spent working at home and my coming home late frustrated and stressed. Her understanding and encouragement gave me the momentum to complete my PhD study.

This thesis has been partially funded by a grant from the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre (DTC).

Contents

Contents	v
List of Figures	ix
List of Tables	xiii
1 Introduction	1
1.1 Focus and Approach of the Research	1
1.2 Thesis Contributions	3
1.3 Organisation of the Thesis	4
2 Ad Hoc Wireless Sensor Networks - Background	7
2.1 Wireless Sensor Networks: Definition and Applications . . .	7
2.2 Challenges and Characteristics of Wireless Sensor Networks .	8
2.3 Hierarchical Routing in Wireless Sensor Networks	10
2.4 Simulation in Wireless Sensor Networks	15
2.5 Chapter Summary	24
3 Related Work	25
3.1 Mapping Applications in the Literature	25
3.2 A Survey on Algorithms for Map Generation	27
3.3 Chapter Summary	30

4	An Approach to Data Extraction and Visualisation for Wireless Sensor Networks	31
4.1	Introduction	32
4.2	Characteristics of Sense Data	33
4.3	The Challenging of Visualisation of Sense Data	35
4.4	Benefits of Sense Data Visualisation	36
4.5	Sense Data Visualisation: Maps	38
4.6	Chapter Summary	39
5	A Mapping Service for Wireless Sensor Networks	41
5.1	Introduction	41
5.2	Understanding the Problem of Map Generation form Sparse Data	42
5.3	A Framework for Mapping Sense Data: Centralised Mapping Service	44
5.4	Chapter Summary	54
6	A Communication Paradigm for the Mapping Service	55
6.1	Introduction	56
6.2	MuMHR Design Objectives	57
6.3	MuMHR Algorithm Details	59
6.4	Optimal Cluster Balancing	69
6.5	Evaluation Metrics	71
6.6	Simulation Experiments	75
6.7	Chapter Summary	87
7	A New Network Service: Map Generation	89
7.1	Introduction	90
7.2	Shepard Interpolation Algorithm Details	90
7.3	Shepard Interpolation Analysis	101
7.4	Chapter Summary	121

8 Distributed Map Generation	123
8.1 Introduction	124
8.2 Distributed Map Construction	126
8.3 Hardware Limitations	135
8.4 Experimental Comparison of Centralised and Hierarchical Mapping Services	136
8.5 Chapter Summary	140
9 An Integrated Inductive-Deductive Framework for Sense Data Mapping	145
9.1 Introduction	146
9.2 Related Work	148
9.3 M-DAD Mapping Framework Details	151
9.4 Distributed Self-Adaptation in the Mapping Service	159
9.5 Experimental Evaluation	165
9.6 Chapter Summary	180
10 Conclusion	183
Bibliography	191
Bibliography	191
A A Survey on Wireless Sensor Networks Simulators	213
A.1 SensorSim	213
A.2 TOSSIM	214
A.3 TOSSF	215
A.4 GloMoSim	215
A.5 OPNET	216
A.6 EmStar	217
A.7 SENS	218
A.8 J-Sim	219

B	A Survey on Cluster-based Routing Algorithms	221
B.1	LEACH	221
B.2	PEGASIS	224
B.3	TEEN and APTEEN	225
B.4	SEP	227
B.5	MESTER	227
B.6	A secure hierarchical model	229

List of Figures

2.1	Simulator usage results from a survey of simulation based papers in MobiHoc 2005 (Kurkowski et al., 2005).	16
5.1	Data is collected at a central point and a global map is computed at once.	45
5.2	Architecture of the centralised mapping service.	47
5.3	The mapping extension of SenSorPlus, the wireless sensor network simulator.	49
5.4	Interpolated field generated from 600 simulated nodes randomly distributed on the surface from 2D map in Figure 7.3	50
5.5	(a) A contour at 116 units drawn over 2D map in Figure 7.3. (b) A contour at 116 units drawn on the interpolated field in Figure 5.4	52
6.1	Routing adaptation to obstacles using the number-of-hops . . .	60
6.2	Network discovery in MuMHR	63
6.3	MuMHR cluster formation process	64
6.4	Handover of cluster head role in MuMHR when a cluster round ends	66
6.5	MuMHR data transmission phase	67
6.6	UML activity diagram that model the entire work flow of MuMHR algorithm.	68
6.7	A 100 nodes wireless sensor network with random topology . . .	75

6.8	Geographically uniform cluster formation.	78
6.9	Node distribution among clusters formed by MuMHR.	79
6.10	A comparison between data delivery ratios of MuMHR and LEACH	81
6.11	Average end-to-end packet delivery delay as a function of node density in the network.	82
6.12	The number of sent messages versus the back-off waiting time in <i>ms</i>	83
6.13	The convergence time versus the back-off waiting time in <i>ms</i> . .	84
6.14	MuMHR network lifetime (first node death)	85
6.15	MuMHR network lifetime (last node death)	86
7.1	Shepard interpolation algorithm bull's-eye effect	93
7.2	Two configurations of co-linear points adopted from (Shepard, 1968).	98
7.3	Grand Canyon height map	103
7.4	Peak profiling with 10 nodes network density	112
7.5	Peak profiling with 50 nodes network density	113
7.6	Peak profiling with 100 nodes network density	114
7.7	Peak profiling with 200 nodes network density	115
7.8	Peak profiling with 300 nodes network density	116
7.9	Peak profiling with 500 nodes network density	117
7.10	Peak profiling with 1000 nodes network density	118
8.1	Architecture of the distributed in-network mapping service. . . .	128
8.2	Local maps are cached and merged at a sink node	129
8.3	A mineral map derived from AVIRIS data.	136
8.4	Interpolated field generated using a centralised mapping service running on 2000 simulated nodes, randomly distributed on the surface from Figure 8.3.	138

8.5	Interpolated field generated using a hierarchical mapping service running on 2000 simulated nodes, randomly distributed on the surface from Figure 8.3.	139
8.6	A contour at 116 units drawn on the interpolated field from Figure 8.4.	140
8.7	A contour at 116 units drawn on the interpolated field from Figure 8.5.	141
8.8	A comparison of centralised and hierarchical mapping based on number of messages sent	142
8.9	Detail section from Figure 8.3	142
8.10	Detail section from Figure 8.4, interpolated using the centralised method	142
8.11	Detail section from Figure 8.5, interpolated using the hierarchical method	143
9.1	Merging of inductive and deductive methods	151
9.2	A Petri-net that models the virtual congress algorithm.	163
9.3	FLIR IinfraCAM SD infrared camera used for taking thermal images of the brass sheet (FLIR Systems, 2008).	168
9.4	A brass sheet with segment hole excavation; Toradex Oak USB sensors; and the experimental apparatus.	168
9.5	Heat diffusion map taken by ThermoCAM P65 Infrared (IR) camera.	170
9.6	Heat map generated by the mapping service using the standard mapping service defined in Chapter 8.	170
9.7	Interpolated heat map generated by M-DAD given obstacle location and length.	171
9.8	Interpolated heat map generated by M-DAD given obstacle location, width and length.	171
9.9	Detailed section of Figure 9.7 and Figure 9.8 showing the area around the obstacle.	172

9.10	Humidity and temperature measurements generated by 10 Toradex Oak USB Sensors.	175
9.11	The Standard Deviation of temperature values at 10 locations calculated by M-DAD using the humidity map.	175
9.12	A network topology that consists of 3 clusters deployed in an area containing an obstacle. In cluster 2 there is only one node which can detect the obstacle. This node detects the obstacle exactly at its end point and generates the best bill in the network. That bill is not going to be agreed in the cluster because the majority of that clusters members are not aware of the obstacle.	179
9.13	Heat map generated by M-DAD with 2cm obstacle length. . . .	180

List of Tables

7.1	(1): 2D maps produced by Shepard and TLI at various network densities.	107
7.2	(2): 2D maps produced by Shepard and TLI at various network densities.	108
7.3	(1): 3D maps produced by Shepard and TLI at various network densities.	109
7.4	(2): 3D maps produced by Shepard and TLI at various network densities.	110
7.5	Peak profiling statistical measures: Mean, Skewness, and Kurtosis	111
7.6	Contour maps drawn on maps produced by Shepard interpolation	120
8.1	Linux-class sensor node hardware platforms.	135
9.1	The obstacle length (in pixels) in the best proposed bill and the agreed bill in three M-DAD mapping runs at 1000 nodes density.	178
9.2	The obstacle length (in pixels) in the best proposed bill and the agreed bill in three M-DAD mapping runs at 100 nodes density.	179
9.3	The obstacle length (in pixels) in the best proposed bill and the agreed bill in three M-DAD mapping runs at 500 nodes density.	179
9.4	The number of bills proposed by the faulty nodes.	180

Chapter 1

Introduction

1.1 Focus and Approach of the Research

Wireless sensor networks offer the potential to give timely information in unattended environments. Dynamic visualisation of wireless sensor network data is needed in order to fully consume that potential, allowing the users to observe spatio-temporal patterns and intrinsic trends in the sensor data. Discovering spatio-temporal patterns in the ever increasing collection of sensor data is an important problem in several scientific domains.

Developing a service that facilitates extraction and representation of sense data presents many challenges. With sensors reporting data at a very high frequency, resulting in large volumes of data, there is a need for a network service that utilises sensor nodes resources efficiently to extract and visualise collected sense data. Although, considerable developments have been made towards effective and efficient data extraction and visualisation of sensory data, these solutions remain inadequate. Most of the solutions available in literature address a specific data extraction or visualisation problem but ignore others (see Chapter 3). None of these solutions is able to simultaneously ensure low energy and memory consumption and provide dynamic (real-time) data extraction and visualisations. Thus, there is a

need for a data extraction and visualisation framework that can balance the conflicting requirements of simplicity, expressiveness, timeliness, and accuracy with efficient resource utilisation. This thesis presents a data extraction and visualisation service, called the mapping service, which solves global network data extraction and visualisation problems through localised and distributed computation.

A service oriented approach has special properties. It is made up of components and interconnections that stress interoperability and transparency. Services and service-oriented approaches address designing and building systems using heterogeneous network software components. This allows the development of a data extraction and visualisation service that works with existing network components, e.g. routing protocols, and resources without adding extra overhead on the network.

After a review of the literature, the research will be approached first by extending a wireless sensor network simulator, Dingo (Mount, 2008), to allow for the development of a simple data extraction and visualisation service. This enabled understanding and analysis of the requirements imposed by such a service onto the nodes and the network infrastructure as a whole. Through identifying and simulating related visualisation applications, such as heat diffusion, challenges that need to be met before a generic service is feasible, will be understood and analysed.

The findings from this stage will be used to produce the specifications for the distributed data extraction and visualisation service. A set of applications, such as isopleth generation, will be characterised as representative problems to expose the key issues in designing a scalable solution for the distributed data extraction and visualisation service and to assess the potential uses of the underlying visualisation information within other network services (such as efficient routing of data).

1.2 Thesis Contributions

The main contribution of this thesis is to develop a distributed data extraction and visualisation service, the *mapping service*. This thesis primarily focuses on the following issue:

- What is a suitable data extraction and visualisation method for sparse wireless sensor network data?
- Is it possible to produce a distributed mapping service that has the following properties:
 - it represents the network as an entity rather than a collection of separated nodes;
 - it is suitable for informing other network services as well as the delivery of field information visualisation;
 - it uses only the existing data and network capabilities of a classical wireless sensor network;
 - it allows the development of applications that are unaware of the reality of sparse data;
 - it accommodates the particular characteristics of the application domain to improve the mapping performance;
 - it combines inductive and deductive methods to map sense data and the universal physical principles;
 - it is suitable for mapping an arbitrary number of sensed modalities;
 - it is capable of mapping one sensed modality from related multiple types of sensed data to overcome the limitations of generating a map from a single type of sense modality.

- What are the requirements imposed by mapping applications on the underlying communication paradigm?
- Is it possible to use the communication architecture to organise the distribution of processing in such a distributed mapping service?
- Is there a map generation method which will provide output maps of a good quality from sparse network data?
 - What are the characteristics of a good map?
 - What is the node density required to generate a good map?
 - What is the cost, in terms of communication and computation, of generating a map at a particular level of accuracy?
- Can these map generation methods be made adaptive and context aware using appropriate multi-variate function fitting techniques?
- By extension, is a distributed mapping service feasible? And if so, are there performance gains from this approach and is it feasible?
 - What are the performance gains of a distributed mapping service over a centralised mapping service?
- What simulation tool is suitable to validate the mapping service against real-world experiments?

1.3 Organisation of the Thesis

The work in this thesis describes progress towards providing a complete distributed data extraction and visualisation service with the capability to exploit the available knowledge about the application domain.

Chapter 2: Ad Hoc Wireless Sensor Networks - Background

This chapter builds upon a basic knowledge of wireless sensor networks. First, wireless sensor networks, their characteristics and design challenges are outlined. Next, an overview of hierarchical routing in wireless sensor networks and the motivation for a new communication paradigm are given. Then, issues related to simulation in wireless sensor networks are discussed and the simulation tools used in this thesis are briefly described.

Chapter 3: Related Work

In the first section, several mapping specific applications in the literature are outlined. The following section describes a number of map generation techniques for wireless sensor networks which utilise a range of network topologies, routing protocols and interpolation algorithms.

Chapter 4: An Approach to Data Extraction and Visualisation for Wireless Sensor Networks

Chapter 4 describes various challenges in data extraction and visualisation in wireless sensor networks. *Map* is proposed as a suitable data extraction and visualisation format. Finally, the challenges in data mapping in wireless sensor networks are shown and maps are shown to fulfil a large set of efficient data gathering and representation requirements.

Chapter 5: A Mapping Service for Wireless Sensor Networks

This chapter describes the mapping service for data extraction and visualisation in wireless sensor networks. A centralised implementation of this service is presented and its merits and shortcomings are discussed. Challenges that need to be met before a distributed service is feasible are identified. Finally, an initial application of the service in the form of isopleth generation is presented.

Chapter 6: A Communication Paradigm for the Mapping Service

In this chapter a new communication paradigm that satisfies a list of requirements imposed by the mapping service are proposed. The new communication paradigm details as well as its performance simulation analysis in both Dingo and NS-2 simulators are provided.

Chapter 7: A New Network Service: Map Generation

Chapter 7 describes a new network service, *map generation*. Map generation is defined as the problem of interpolation from sparse data collected by a wireless sensor network. Two interpolation methods, Shepard interpolation and Triangulation with Linear Interpolation are investigated. Finally, the suitability of Shepard interpolation for wireless sensor networks is illustrated through extensive experimentation using real data.

Chapter 8: Distributed Map Generation

Chapter 8 is the main exposition of the distributed map building algorithm. It explains how to exploit other network services, such as the communication and map generation services, to provide more efficient, yet, less expensive mapping solutions. Finally, the distributed mapping service performance simulation results are compared to those of the centralised mapping service implemented in Chapter 5.

Chapter 9: An Integrated Inductive-Deductive Framework for Sense Data Mapping

This chapter elevates the capabilities of the distributed mapping service to improve the produced map quality. The improved mapping service is shown to be capable of generating maps from an arbitrary number of dimensions. Next, it is illustrated how the mapping performance can be enhanced by exploiting knowledge about the application domain to adapt the mapping service behaviour. Finally, an algorithm that is used to maintain up-to-date knowledge of the current state of the application domain is presented. The experimental results obtained here are compared to those in chapters 5 and 8.

Chapter 10: Conclusion

Conclusions are drawn and further work suggested.

Chapter 2

Ad Hoc Wireless Sensor Networks - Background

2.1 Wireless Sensor Networks: Definition and Applications

An Ad Hoc Wireless Sensor Network generally consists of a number of low-cost devices that are spatially distributed across a geographical area to measure the ambient conditions of a monitored phenomenon. Typically, a sensor node has a wireless communication capability, processing unit, memory, sensors, and power source. Individual nodes are characterised by limited resources; but when their information is coordinated with the information from a large set of other nodes they become capable of monitoring a physical environment in great detail.

Wireless sensor networks have several advantages. The low cost and small size of sensors make such networks applicable in many applications. They are self-organising and easy to deploy. They cover large areas. They may have some level of fault tolerance which maintains the network operation when one or few nodes fail. They have close connection with their

environment without causing disturbance to that environment. They can be an economical method for long-term data gathering and they help to avoid unsafe or unwise repeated field studies. Finally, these networks can operate unattended in potentially hostile or hazardous environments. All of these advantages among others make wireless sensor networks appropriate in a variety of fields such as environmental monitoring, in the military, health care, agriculture, commerce, civil engineering, surveillance, etc. For example, in health care applications, wireless sensor networks can be used to monitor disabled people and biomedical sensors can help to create vision; environmental applications may range from habitat monitoring to environment observation and forecasting systems, e.g. using thermal sensors to monitor temperature in a forest; commercial applications may range from managing inventory, monitoring and controlling machines to monitoring product quality. Military applications range from target tracking to enemy movement detection in the battlefield.

Wireless sensor networks have been recognised as one of the most important technologies for the 21st century. Exploiting the potential of wireless sensor networks will provide efficient and cost effective solutions to many problems (Arboleda and Nasser, 2007). They hold a lot of promise in applications where gathering sensing information in remote locations is required.

2.2 Challenges and Characteristics of Wireless Sensor Networks

Unlike traditional communication networks, wireless sensor networks generally have no global identification scheme, due to the large number of nodes and the high overhead required for ID maintenance. They mainly use the broadcast communication paradigm to transmit sensed data to the required destination. Sensor nodes require careful usage as they are limited in resources. Many sensor nodes are battery powered which makes their life

dependent on the battery life. Power consumption is mainly spread over three tasks: communication; sensing; and processing. According to Pottie and Kaiser (2000), communication consumes more energy than processing. Power conservation is acquiring more importance, and researchers are paying more attention on designing power aware sensor networks.

Scalability is one of the important features of a good network design (Alazawi et al., 2008). Some sensor network applications require a dense node deployment in order to enhance the network reliability and the accuracy of collected data. In such dense networks, sensors may have overlapping coverage areas (Deng et al., 2005). Underlying network routing protocols should be able to operate correctly with dense network deployments and exploit node redundancy in saving energy, distributing load, and fault correction. Data collected in overlapped areas will be highly correlated and redundant (Anastasi et al., 2009). Since sensor nodes may generate significant redundant data, intermediate nodes may apply message aggregation to reduce the number of transmissions (Alzaid et al., 2008). Because the number of sensor nodes may exceed thousands in some applications, the cost of the sensor node has to be kept low (Beigl et al., 2005).

Wireless sensor networks are fault-prone and since their on-site maintenance is not always feasible, self-healing and robustness is essential for enabling the deployment of large-scale networks (Turau and Weyer, 2009; Chang et al., 2006).

Wireless sensor networks may operate in remote hostile environments; they can be deployed under the soil, under water, in a biologically contaminated field, in a building (Chorzempa et al., 2007) or embedded in machines. These networks are also characterised by unreliable wireless communication channels which poses a challenge of how to provide reliable transmission over wireless links (Akyildiz et al., 2002). The wireless transmission medium can take many forms including radio, infrared, or optical media.

2.3 Hierarchical Routing in Wireless Sensor Networks

Routing has proved to be a key issue in this research. Based on the literature review (Ganesan et al., 2003), it is well-understood that the performance of the data extraction and visualisation service often depends profoundly on efficient and reliable communication. However, it is difficult to attain both scalable and robust communication in ad-hoc wireless sensor networks. As clustering approaches are particularly tempting to large-scale high-density sensor network applications, a solution from the notion of clustering for ad hoc wireless communications is sought.

Many new communication protocols have been specifically designed for sensor networks in which energy awareness is an essential consideration (Yang et al., 2006; Babbitt et al., 2008a). Most of the interest, however, has been given to the routing protocols since they might differ depending on the application and network architecture. Hierarchical routing is one of the most popular routing schemes in wireless sensor networks (Heinzelman, 2000; Lindsey et al., 2001; Manjeshwar and Agrawal, 2001, 2002; S.Lindsey and Raghavendra, 2002; Smaragdakis et al., 2004; Ye et al., 2005). It is a two or more tier routing scheme known for its scalability and communication efficiency. Nodes in the upper tier are called cluster heads and act as a routing backbone, while nodes in the lower tier perform the sensing tasks. Kulkarni et al. (2005) argue that multi-tier networks are scalable and offer the following number of advantages over single-tier networks: lower cost, better coverage, higher functionality, and better reliability. A sink-based single tier network can lead to congestion at the gateway especially in dense sensor networks. This can cause communication delays and inadequate tracking of the sensed events. Moreover, some of the routing algorithms for such network architecture are commonly not scalable, e.g. (Muruganathan et al., 2005). To overcome these problems, network clustering has been proposed

as a possible solution.

Many clustering algorithms have been previously investigated in the context of routing protocols or independently of routing protocols (Basagni, 1999; Bandyopadhyay and Coyle, 2003; Amis et al., 2000; Banerjee and Khuller, 2001; Perkins and Royer, 1999). In Appendix B, some of the most prominent cluster-based routing algorithms in the literature are surveyed and their relative advantages and disadvantages are discussed.

Motivation for a New Routing Protocol

After some of the existing routing algorithms and their characteristics have been considered, it is concluded that a routing protocol that satisfies many important data extraction and visualisation, network, and application requirements is missing.

Low Energy Adaptive Clustering Hierarchy, or *LEACH* (Heinzelman et al., 2000) and its derivatives have been successful in reducing the energy per bit required by each node and the network as a whole to communicate from the nodes to the sink. Nonetheless, most of these protocols are built upon inflexible assumptions and have serious drawbacks. Many clustering algorithms are mostly heuristic in nature and have a time complexity of $O(n)$ (Heinzelman, 2000; S.Lindsey and Raghavendra, 2002; Heinzelman et al., 2000), where n is the total number of nodes in the network. Also many protocols, such as that discussed by Smaragdakis et al. (2004), demand time synchronisation among nodes which makes them unsuitable for large-scale networks. Another set of protocols requires centralised management which limits their scalability. Some of these protocols have been designed with robustness in mind. However, the level of fault tolerance is usually designed to adapt to occasional node failures and infrequent topology migration.

In order to support various computationally demanding applications for large-scale wireless sensor networks, such as the mapping applications (Shuttleworth et al., 2007), a routing protocol that satisfies the following require-

ments is needed:

1. Real-time requirements: A good routing protocol is required to provide timely communication by reducing end-to-end communication delays. Timeliness constraints are important as sensor networks operate in the real world to reflect the physical status of the sensing environment. Therefore, the sensor data is valid only for a limited time and hence needs to be delivered within certain time bounds. For example, the location data for a moving target requires real-time data delivery to be useful for producing an effective response. Data timeliness is normally at odds with energy consumption (Abdelzaher et al., 2004). Data aggregation schemes delay delivery until more data can be aggregated to reduce total network traffic. However, aggregation will reduce temporal performance due to the introduced aggregation delay.

Timely data delivery also has a significant effect on information visualisation and extraction services. As visualisation aims to help scientists to quickly interpret data and make decisions, the key advantage of timely visualisation is that the event of interest is still in process and it is not too late to take additional action. Consequently, timeliness issues have to be addressed to meet end-to-end delivery deadlines with acceptable energy consumption levels.

2. Reliability requirements: The intended routing protocol should be reliable and robust to message loss. Wireless sensor networks are exposed to several faults mainly due to unreliable communication channels and sensor node failures. However, routing protocols must find alternative routes to the destination and sustain the overall function of the sensor network system. A reliable and robust routing protocol is to be capable of providing correct measurements at the right moment without interruption. Different wireless sensor network ap-

plications have varied requirements on the reliability of data delivery. These requirements may evolve over time. For example, a wireless sensor network deployed for fire detection in a forest can be used for measuring the humidity as well. When the measured temperature is in the range of normal temperature it is delivered to the sink tolerating a certain percentage of loss. Yet, when a fire is detected, the data should be delivered to the sink with high priority. Given the sensor network dynamics, the routing protocol must support the ability of the network to ensure reliable data transmission in the state of continuous change of network topology.

3. Scalability requirements: Scalability is the ability of the network to grow, in terms of the number of nodes, without excessive communication overhead. To improve scalability, some routing protocols utilise multi-hop communication. In wireless sensor networks it becomes more difficult to build a reliable network as the number of nodes increases due to the network management overhead that comes with the increased network size (Alazzawi et al., 2008; Deng et al., 2005; Smaragdakis et al., 2004). In a larger scale wireless sensor network more overhead is unavoidable to keep the build of the communication paths to the sink. Since the available overall bandwidth is limited, the increase of overhead results a smaller amount of bandwidth left for application data transmission. Therefore, routing protocols should support in-network processing, such as lossy in-network data aggregation (Abdelzaher et al., 2004), to reduce the amount of communicated data to save bandwidth utilisation and energy. Besides being able to operate correctly with large numbers of nodes, the required routing protocol must also be able to exploit the density of sensor nodes in saving energy, distributing load, and fault correction.

4. Energy utilisation requirements: wireless sensor networks are limited

in physical resources mainly memory, processing, energy and bandwidth. The desired routing protocol must minimise the total energy expenditure and prolong network lifetime because sensor nodes are usually powered by batteries which limits the life of a sensor node. Thus, the routing protocol must use the resources efficiently to carry out data communication tasks.

5. Load-balancing requirements: One of the main challenges in sensor networks routing protocols design is balancing the resource usage of individual nodes with the desired global network behaviour. The desired routing protocol must be able to reduce the communication and processing overload as well as distribute energy requirements to achieve longer network life. Distributing workload amongst nodes will also help to prevent energy depletion in one part of the network.
6. Clustering requirements: Clustering must be kept simple and decentralised. Each node should be able to independently make its decisions based on the available local information. A distributed implementation of clustering algorithms is expected to create minimal communication overload on the sensor nodes. Clusters setup should be efficient in terms of processing and communication. Furthermore, the clustering algorithm should assist achieving load-balancing requirements through fair distribution of sensor nodes among various clusters. Also, the total number of clusters in the sensor network should be sustained equal or around the optimal number of clusters defined by Heinzelman et al. (2000) which is 5% of the total number of nodes in the network.
7. Location information: The desired routing protocol should be able to setup the network into clusters and balance load among clusters without location information. However, it should be open to function with location information if the application requires so.

2.4 Simulation in Wireless Sensor Networks

A successful large-scale wireless sensor network deployment necessitates that the design concepts are checked before they are optimised for a specific hardware platform. Developing, testing, and evaluating network protocols and supporting architectures and services for wireless sensor networks can be undertaken through test-beds or simulation. Whilst test-beds are extremely valuable, implementing such test-beds is not always viable because it is difficult to adapt a large number of nodes in order to study the different factors of concern. The substantial cost of deploying and maintaining large-scale wireless sensor networks and the time needed for setting up the network for experimental goals makes simulation invaluable in developing reliable and portable wireless sensor networks applications.

In wireless sensor networks, simulation provides a cost effective method of assessing the appropriateness of systems before deployment. It can, for example, help assess the scalability of algorithms free of the constraints of a hardware platform. Furthermore, simulators can be used to simplify the software development process for a particular wireless sensor network application. For instance, TOSSIM (Levis et al., 2003) utilises the component-based architecture of TinyOS (Levis et al., 2005) and provides a hardware resource abstraction layer that enables the simulation of TinyOS applications which can then be ported directly into a hardware platform without further modifications.

Simulation is hence the research tool of choice for the majority of the mobile ad hoc network community. According to Kurkowski et al. (2005), 75.7% of the authors in the premiere conference MobiHoc 2005 (ACM, 2007) used simulation in their research. Apart from the self-developed simulators, there are a few widely accepted network simulators including NS-2 (NS-2, 2007), GloMoSim (Zeng et al., 1998), TOSSIM (Levis et al., 2003), and OPNET (OPENET Technologies Inc, 2007). Figure 2.1 adopted from (Kurkowski et al., 2005) shows the simulator usage following a sur-

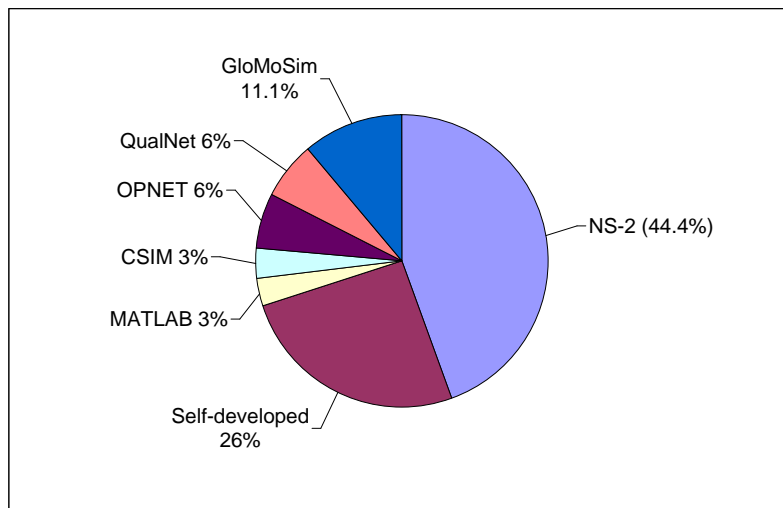


Figure 2.1: Simulator usage results from a survey of simulation based papers in MobiHoc 2005 (Kurkowski et al., 2005).

vey of simulation based papers in MobiHoc. Simulation of ad hoc wireless capabilities for wireless sensor networks have been addressed by extending existing simulators, as in the NS-2 project, or specifically building new ones, such as J-Sim (Kačer, 2002). The latter class of simulators mostly focuses on protocols and algorithms for layers of the network stack, but they do not directly support sensor networks.

Recently, several simulation tools have appeared to specifically address wireless sensor networks, varying from extensions of existing tools to application specific simulators. Although these tools have some collective objectives, they obviously differ in design goals, architecture, and applications abstraction level. In the next subsection, we review some of the important wireless sensor networks simulation tools and explore their characteristics.

Simulation Tools

In this work we use simulation to implement and evaluate various proposed protocols and services. Appendix A is devoted to a study of wireless sen-

sor network simulators that focus on wireless protocol stacks for sensor networks, such as NS-2 (NS-2, 2007), SensorSim (Park et al., 2000), SensorSimII (Ulmer, 2007), TOSSIM (Levis et al., 2003), TOSSF (Perrone and Nicol, 2002), GloMoSim (Levis et al., 2003), and OPNET (OPENET Technologies Inc, 2007). However, the following subsections study, in detail, the simulators that were used in this work.

NS-2

NS-2 (NS-2, 2007) is an object-oriented discrete event simulator targeted at networking research. It is an open source network simulator originally designed for wired, IP networks. NS-2 provides substantial support for simulation of various network protocols over wired, wireless, and satellite networks. All NS-2 simulations are implemented using a combination of C++ and Tcl. C++ is used to implement the main functionality of NS-2 such as extending its libraries, while the dynamics of the simulation are controlled by Tcl scripts. The fundamental components in NS-2 are the layers of the protocol stack. NS-2 has a simulation event scheduler, network component object libraries, and network setup module libraries.

There have been many efforts to extend NS-2 to simulate wireless sensor networks such as SensorSim (Park et al., 2000) and SensorSimII (Ulmer, 2007). Support is included for many of the unique features of wireless sensor networks such as sensing channels, sensor models, battery models, various protocol stacks for wireless sensors, hybrid simulation support, and scenario generation components. An extension developed in 2004 by Downard (2004) allows for external phenomena to trigger events. Currently, the NS-2 simulation environment offers great flexibility in studying the characteristics of sensor networks because it includes flexible extensions for wireless sensor networks. In the NS-2 environment, a sensor network can be built with many of the same set of protocols and characteristics as those available in the real world such as the IEEE 802.11. The extensions also include

support for node movements and energy constraints. According to (Curren, 2005), NS-2 is the most popular simulation tool for sensor networks for the following reasons:

1. It has modular design which has effectively made it extensible.
2. The object-oriented design of NS-2 allows for straightforward creation and use of new protocols.
3. It supports several routing and queueing algorithms.
4. A high number of different protocols are publicly available, despite not being included as part of the simulator's release.
5. It has the capability to generate graphical details of network traffic through the Network Animator (NAM).
6. Its status as the most used sensor network simulator has also encouraged further popularity, as developers would prefer to compare their work to results from the same simulator.

As with many wireless sensor network simulators, NS-2 has a number of limitations. First, NS-2 puts some restrictions on the customisation of packet formats, energy models, MAC protocols, and the sensing hardware models which limits its flexibility. Second, the lack of an application model makes it ineffective in environments that require interaction between the application level and the network protocol level. Third, NS-2 does not run real code; this is an important drawback if the purpose is to deploy code or use simulation to track down a bug observed in code that has already been deployed. Moreover, NS-2 does not have the notion of a phenomenon such as moving vehicles that could trigger nearby sensors through a channel. Finally, despite its capabilities, NS-2 has been built by many developers and contains several inherent known as well as unknown bugs. Furthermore,

NS-2 does not scale well for sensor networks partially due to its object-oriented design (Curren, 2005). Object oriented design is favourable in terms of extensibility and organisation but it causes performance bottlenecks in large-scale sensor network environment. Every node has its own object and can interact with every other node in the simulation, generating a large number of dependencies to be checked at every simulation interval. NS-2 is capable of handling up to 16,000 nodes, but the level of detail of its simulations leads to a running time that makes it desperate to deal with more than 1,000 nodes on a standard PC. The highly detailed packet level simulations lead to a run-time behaviour closely coupled with the number of packets that are exchanged, making it unsuitable for simulating large-scale networks.

SenSor

Mount et al. (2005) developed a scalable Python based simulator, called SenSor, that provides a workbench for prototyping algorithms for wireless sensor networks taking a top-down design methodology. This design methodology has been found to be of particular value in developing algorithms by refinement. Having no target platform means the full functionality of a programming language can be used. This eases the design process as prototype algorithms can be tested before optimisation for the target platform (Liarokapis et al., 2007). Its design reflects the cross-domain nature of the wireless sensor network field and the consequent mix of concerns and requirements of end-users in a real-world sensing application. SenSor consists of a fixed API, with customisable internals. It has a simple graphical user interface and a set of base classes which are extended by the user to create simulation. Although an object oriented approach has been used in the design, the base classes are loosely coupled. The SenSor Graphical User Interface (GUI) has the following features:

1. Built-in code editor with syntax colouring and folding

2. A graph pane on which a representation of the sensor network is drawn
3. A chart pane where individual data can be plotted during the simulation

Each simulated sensor node runs in its own thread and communicates using the same protocols as its physical node would be. This enables experimentation with different algorithms for managing the network topology, simulating fault management strategies and so on, within the same simulation. Sensors are modelled using a pool of concurrent, communicating threads. Individual sensors are able to:

1. Gather and process data from a model environment
2. Locate and communicate with their nearest neighbours
3. Determine whether they are operating correctly and act accordingly to alter the network topology in case of faulty nodes being detected

Nodes may be configured differently to simulate a heterogeneous sensor network. This is particularly useful when different nodes have different capabilities, or to test the effect of deploying new sensor nodes into an existing network. Another good feature in SenSor is that it comes with a set of application level routing packages including simple multi-hop flooding, Directed Diffusion, Rumor Routing, and LEACH.

As with SensorSimII, SenSor provides an extensible visualisation framework which aims at easing the life for sensor network debugging, assessment, and understanding of the software by visualising the sensor network topology, the individual node state, and the transmission of the sensed data. This graphical interface was created to allow visual interaction with a sensor network. A few existing wireless sensor network simulation tools, such as J-Sim, cover some aspects of sensor network data display and visualisation. To aid debugging, colour can be used to represent the state of each

node and the type of packets sent across the network. In SenSor, nodes are represented by circles and communication is visualised by arcs drawn from the sending node to the receiving node. For example, in MuMHR simulation experiments (Section 2.3), nodes in each cluster are assigned a distinctive colour to show the nodes distribution among clusters. Users may drag and drop nodes around the pane which automatically changes the internal representation of their geographical location. This is particularly useful for experimenting with localisation algorithms. Furthermore, separate interfaces gather information from the network and display it on the graph pane or the chart pane, where individual data can be plotted during the simulation. This partitioning allows users to experiment with different ways of processing individual node data into information. When a node has been selected, its data is displayed on the chart pane.

SenSorPlus

SenSorPlus (Liarokapis et al., 2007) is a further extension of SenSor with an added interface between the simulation environment and different hardware platforms, for example the Gumstix (Gumstix.com, 2007) platform. SenSorPlus bridges between SenSor and the hardware to allow the same source code that is executed on simulated sensor nodes to also be deployed on actual sensor nodes; this enables application portability. SenSorPlus users write high level simulations, which can then be refined to various hardware platforms. In addition, SenSorPlus allows mixed mode simulation using a combination of real and simulated nodes. Initial evaluations proved that it was not only easy to use, but also powerful enough to model and simulate the behaviour of a system at various design stages. It provides an easy way to develop system models, enabling users to quickly manipulate hardware elements and achieve the desired results without having to build a full hardware prototype. This extension reduces the development time and lessens the scope for errors being introduced. In SenSorPlus, nodes have the ability

to obtain their sensed data from a database or graphical objects like maps. The use of real-world data improves the fidelity of simulations as it makes it possible to check the simulation results against the real data.

SenSorPlus focuses on the protocols and algorithms for higher layers of the network stack but does not directly support sensor networks physical layer. It has scalability bounds due to its object oriented design. Furthermore, it has major drawbacks which limit its functionality. Most of these drawbacks are due to the incomplete nature of the tool. These drawbacks are:

1. The lack for Media Access Control or *MAC* layer: SenSorPlus does not support any MAC protocols, such as Carrier Sense Multiple Access or *CSMA*. Communications are handled by point-to-point systems.
2. No collision management procedure: Partly due to the absence of the MAC layer, SenSorPlus ignores the packet collisions aspects that occur in real-world wireless sensor networks. Packet collisions simulation adds more realism to test for algorithms robustness. SenSorPlus uses a Scheduler API to schedule execution of a list of send and receive tasks. The Scheduler API only runs one task at a time leaving no chance for collisions to happen. This execution mechanism does not model collisions which is a critical issue in real-world sensor networks deployments.
3. No environment model: Wireless sensor network applications attribute integration of communication, computation, and interaction with the physical environment. The environment component models the characteristics of an environment which has effect on the functionality of a sensor network such as network propagation characteristics.
4. Energy model: SenSorPlus does not provide any support for simulating energy consumption in wireless sensor networks. This makes

SenSorPlus unsuitable for measuring the network life-time or testing for algorithms energy efficiency.

5. Simulator performance: In SenSorPlus simulations, a large number of simulated nodes could lead to performance bottlenecks because each simulated sensor node runs in its own thread. This threading scheme causes dramatic loss of efficiency because of the context switching and the physical limit on the processing power of the simulation running machine.

Dingo

Dingo (Mount, 2008) is a development tool for wireless sensor networks. It is a fork of the SenSorPlus project in development at the University of Wolverhampton, UK. Dingo features several improvements on SenSorPlus and other simulators. Most importantly, Dingo features a new customisable energy module which makes it more flexible than the energy model used by NS-2 and other simulators. Moreover, Dingo features a significant improvement in the simulation performance by giving the option to split the visualisation from the simulation. It forces thread switching to occur so that threads can not dominate the scheduler which makes the GUI more responsive.

Dingo provides more tools than SenSorPlus for the simulation and deployment of high-level, Python code on real sensor networks. For example, Dingo-boom provides a two-way interface between MoteIV's Boomerang class motes and Dingo. Dingo-top is another tool which is used to dump network topology data to a text file and generate a graphical representation of that topology. Furthermore, Dingo has several new features in the form of plugins. These can be activated/deactivated on the plugin menu. As in TOSSIM, Dingo has a "Topology" menu which can be used to change the network topology of a simulation from a random topology to/from a grid. Network topologies can be loaded and saved.

2.5 Chapter Summary

This chapter has briefly overviewed the characteristics of wireless sensor networks, challenges facing application developers, communication paradigms, and their simulation tools. Based on the survey of the existing hierarchical routing algorithms, it is proposed that a new hierarchical routing algorithm that satisfies a defined list of sense data extraction and visualisation service requirements is needed. Finally, in the presence of a large number of wireless sensor networks simulators, a detailed survey of various simulation tools was performed to find an adequate simulator for testing and evaluating the work in this thesis.

Chapter 3

Related Work

3.1 Mapping Applications in the Literature

Within the wireless sensor network field, mapping applications found in the literature are ultimately concerned with the problem of mapping measurements onto a model of the environment. Estrin (2007) proposed the construction of isobar maps in sensor networks and showed how in-network merging of isobars could help reduce the amount of communication. Furthermore, Meng et al. (2006) proposed an efficient data-collection scheme, and the building of contour maps, for event monitoring and network-wide diagnosis, in centralised networks. Solutions such as Distributed Mapping have been proposed to the general mapping domain. However, many solutions are limited to particular applications and constrained with unreliable assumptions. The grid alignment of sensors in (Estrin, 2007), for example, is one such assumption.

In the wider literature, mapping was sought as a useful tool in respect to network diagnosis and monitoring (Meng et al., 2006), power management (Tynan et al., 2005), and jammed-area detections (Wood et al., 2003). For instance, contour maps were found to be an effective solution to the pattern matching problem that works for limited resource networks (Xue et al.,

2006). Rather than resolving these types of isolated concerns, in this work the wireless sensor network is expected not only to produce map type responses to queries but also to make use of the data supporting the maps for more effective routing, further intelligent data aggregation and information extraction, power scheduling and other network processes.

These are examples of specific instances of the mapping problem and, as such, motivate the development of a generic distributed mapping framework, furthering the area of research by moving beyond the limitations of the centralised approaches.

Work already undertaken by the author, (Shuttleworth et al., 2007), presents a first step towards a mapping service framework, with an example application built upon it. Although a simple approach was taken, the feasibility of implementing a sophisticated mapping service was assessed and an opportunity identified to investigate its usefulness.

Leading directly from this work, there are a number of avenues that need to be followed. Firstly, the distributed mapping service would rely on energy efficient data collection, especially when mapping constantly changing modalities (i.e. the field parameter to be mapped changes within the range of sensing modalities the nodes are capable of). Further, the ability to produce representations at arbitrary scales and foci relies on powerful querying and semantic groups' definition within the network, which is a current research challenge for the wireless sensor network community. A further avenue that has had little attention in the literature and needs more investigation is the application of mapping services to other network exposures such as barriers that might cause discontinuities in the map or applications built upon it. For example, other services such as JAM (Wood et al., 2003) that integrate entirely into the network system software could be moved to a more suitable, higher level if they were built upon the underlying mapping service.

3.2 A Survey on Algorithms for Map Generation

The following section provides a brief on related work in map generation methods. Map generation techniques have previously been explored in the context of wireless sensor networks (Xue et al., 2006; Chang et al., 2006). Chang et al. (2006) implemented an algorithm to estimate sensor nodes faulty behaviour, e.g. faulty readings due to environmental interferences, on top of a cluster-based network. In this work, each cluster head has a fault rate table to store the fault-estimation rate of each sensor node attached to it. The cluster head estimates the fault rate every time it receives a new packet and compares it with a defined threshold; if the new fault rate of the node is greater than the threshold, it will be stored into the fault rate table. Finally, the fault map is constructed using the up to date fault estimation rate table. This approach is based on Bayesian Belief Networks (BBNs) which make it problematic to compute all the probabilities and the revised probabilities once a new sensor reading is received. In high density networks and when nodes can sense more than one modality, the number of dependencies increases rapidly and probabilities computation becomes an NP-hard problem. This approach also lacks precision when updating the fault rate table since it is based on a predefined threshold value.

Event detection based on matching the contour maps of in-network data distribution has been shown effective for event detection in wireless sensor networks (Xue et al., 2006). This event based detection technique was based on the observation that events in sensor networks can be abstracted into spatio-temporal patterns of sensory data and pattern matching can be done efficiently through contour map matching. Using these contour maps as building blocks, events based on the spatio-temporal patterns exhibited in the contour maps are defined. The contour map is constructed hop-by-hop from bottom up in the network as a special aggregation function instead of

transmitting the data to a central point to construct the map centrally. In this map construction scheme, a rectangular $m \times n$ grid with the square cell length i is imposed on the network topology and a multi-path, ring-based routing scheme is adopted for data transmission. Each cell of the grid can have at most one node inside. The data is received by and processed on every neighbour of the node that is one hop closer to the sink node. The aggregated data generated and transmitted by a node is the contour map of a sub-network rooted at the node. This data is defined as a *partial map*. A partial map of a node consists of a set of disjoint contour regions where each contour region is an orthogonal polygon in the two dimensional plane in the grid setting. A two dimensional polygonal contour is stored as a linked list in a partial map. The construction starts from each node generating a partial map of its own. When a node in the transmission line to the sink receives data from its neighbours, it adds each contour region in these partial maps with its own, to produce a new set called the *working partial map*. This process is repeated until the final partial map of the node is generated. This approach works well with grid network topologies and less well with random topologies, which may be more common in real life applications. When a grid is overlaid on top of a random topology some cells in the grid may be empty. These empty cells will not participate in the final partial map construction. Hence, the final map will not cover the entire network area. This makes the scheme sensitive and unsuitable for random sensor networks deployments. Furthermore, the loss of any partial maps will result in an incomplete network map.

In both (Xue et al., 2006) and (Chang et al., 2006) the sink node is required to know the location and the ID of all nodes in the network. Furthermore, the work in both papers is not application-independent and requires major lower level modifications if the application is to change. In (Chang et al., 2006), it is not clear how the hierarchy is built. Besides, it is only suitable for small size networks due to the single hop communication scheme.

In (Xue et al., 2006), the assumptions made on the network topology and the way the grid is formed are not efficient and may dissipate the energy savings achieved by the in-network map construction.

The distributed mapping service, proposed in Chapter 8, has been partly inspired by the work of (Ganesan et al., 2003). Their work made the case for a large-scale distributed multi-resolution storage system that provides a unified view of data handling in sensor networks incorporating long-term storage, multi-resolution data access and spatio-temporal correlations in sensor data. This work is related to ours, but different in focus at both the system architecture and coding level. It outlines an approach for relatively power-rich devices, focused on encoding regularly-gridded, spatial wavelets over time series. For data transmission, Ganesan et al. (2003) designed a routing protocol, wavRoute, which minimised the communication overhead and balanced communication, computation and storage load. wavRoute used a recursive grid decomposition of a physical space into tiles which makes it unsuitable for random topology networks. By contrast, we focus on highly resource constrained devices, and integrate the routing and map generation services (see Chapter 7) with the mapping service. Our work is also focused on spatially deployed networks and is independent on a particular routing or interpolation algorithm.

In centralised map generation approaches, delivering all network sensory data back to the sink incurs heavy transmission traffic, which quickly depletes the energy of sensor nodes and causes bandwidth bottlenecks. Several aggregation based map generation methods have been proposed to address this problem (Shuttleworth et al., 2007; Meng et al., 2006; Xue et al., 2006; Zhao et al., 2002). However, aggregation based methods can not further improve the scalability of the network as all sensors are required to report to the sink. Moreover, the aggregation process increases the computation overhead on the intermediate nodes. To address the inherent limitations of aggregation based methods, Liu and Li (2007) proposed a method called

Iso-Map that intelligently selects a small portion of the nodes, *isoline nodes*, to generate and report mapping data to reduce the network traffic and computation overhead. Partial utilisation of the network information leads to a decrease in the mapping fidelity and isoline nodes will suffer from heavy computation and communication load. Furthermore, the location of mapping nodes can also affect the directions of traffic flow and thereby have a significant impact of the network lifetime. Finally, in sparsely deployed low density networks it is difficult to construct contour maps based only on isoline nodes. The positions of isoline nodes provide only discrete *iso-positions* which does not define how to deduce how the isolines pass through these positions.

To conclude, mapping is often employed in wireless sensor network applications but as yet there is no clear definition (or published work towards) a distributed mapping service architecture that would aid the development of more sophisticated services. The development and analysis of such a service is the key novel contribution of the work proposed here.

3.3 Chapter Summary

While current research has made headway in the sense data mapping area, many problems remain unsolved. Based on the survey of mapping related applications as well as map generation techniques and observations made regarding the efficacy of these algorithms, it is proposed that only a *distributed* mapping service can provide adequate data visualisation and extraction in wireless sensor networks.

Chapter 4

An Approach to Data Extraction and Visualisation for Wireless Sensor Networks

Ever since Descartes introduced planar coordinate systems, visual representations of data have become a widely accepted way of describing scientific phenomena. Modern advances in measurement and instrumentation have required increasingly sophisticated visual representations, to ensure that scientists can quickly and accurately interpret increasingly complex data. Most recently, wireless sensor networks have emerged as a technology which is capable of collecting a vast amount of data over space and time. This data is usually saved in the form of numerical data in a central repository. Often, the eventual aim is to derive an estimate of a parameter or function from this data. The sheer volume of data makes it difficult to be interpreted by humans in meaningful ways. Visualisation techniques help to turn large amounts of raw data into credible visual information such as graphs, charts, or maps that can assist in drawing conclusions from the data. Visualisation is the process of representing abstract data as images using image processing and computer graphics techniques to identify patterns and anomalies, and

to generate further hypotheses (Birdsong and Helms, 2007). In this chapter we discuss challenges for developers of visualisation techniques which seek to “map” the data sensed by a wireless sensor network.

4.1 Introduction

The main objective of a sensor network is to provide users with access to the information of interest from data gathered by spatially distributed sensors. Typically, these networks are deployed to provide useful insights regarding the state of the monitored environment, as opposed to providing a set of raw data readings (Giridhar and Kumar, 2006; He et al., 2004). In many real-world applications, wireless sensor networks are deployed in a high density to ensure a full coverage of the monitored phenomena. These networks are expected to generate very large amounts of data. As the sensor network scales in size, so does the amount of sensed data which is required to be collected by the network, processed, and presented to the user. The data produced and the form in which they are structured varies widely. Scientists need tools to reconcile these differences. Human interpretation of this data would require extensive use of visualisation tools. Using scientific visualisation techniques can help to make meaningful visualisation possible in many aspects of data understanding and analysis. Data visualisation is becoming increasingly important in wireless sensor networks as it enables end-users to best utilise the data collected by the sensor network.

This chapter examines the problem of sense data visualisation. The challenges which it is aimed to overcome are outlined and an overview of various sense visualisation related topics is provided followed by the definition of a suitable data visualisation format.

4.2 Characteristics of Sense Data

Data acquired from a wireless sensor network is imperfect in nature. This imperfect nature of data is due to physical constraints on node deployment and data collection, a noisy environment, device measurement errors, among other factors. Moreover, data collected by different sensors may have various qualities depending on physical characteristics such as distance from sensed phenomena, sense modality, or noise model of individual sensors (Kang and Li, 2006).

In densely deployed wireless sensor networks, sensor readings are usually highly correlated in the space domain (Akyildiz et al., 2004). Additionally, the nature of the physical phenomenon contains the temporal correlation between successive readings of each sensor node (Liu and Li, 2007). Data gathered from a wireless sensor network is often characterised by its significant redundancy. Many sensor networks are densely deployed with high node redundancy to deal with node failure based connectivity and coverage problems. However, dense deployment causes neighbouring sensor nodes to have highly overlapping sensing regions. Consequently, it is likely that multiple nodes often detect and communicate data packets about common phenomena. This is due to the fact that each node observes the physical region of overlap independent of its neighbours. Data aggregation techniques (Abdelzaher et al., 2004; Chan et al., 2006; Solis and Obraczka, 2003; Lee and Chung, 2004) aim at reducing redundant data transmissions. Although data aggregation results in fewer transmissions, it introduces a considerable amount of delay in delivering the data to its final destination. Data from nearer sources may have to be held back at an intermediate node in order to be aggregated with data coming from sources that are farther away. In addition, within such networks, delays may be caused by hop-by-hop retransmission, scheduled data communication, queueing delays, propagation through the environment, and other factors.

In many wireless sensor networks application areas, such as medical

or surveillance applications, the accuracy of acquired data is often crucial. However, sensed data is often inaccurate and erroneous (Solis and Obraczka, 2003). This inaccuracy may be a result of faulty sensor readings, internal errors in sensor nodes, network delays, amongst others. The deployment of a larger number of sensor nodes provides potential for greater accuracy in the information gathered. The ability to effectively increase the sensing quality, without necessarily increasing data transmissions, will increase the reliability of the information for the end user application. Some schemes such as that discussed by Sugihara and Chien (2005), trade data accuracy for energy efficiency, which typically increases with the amount of data transmissions. Data aggregation also increases the level of gathered data accuracy and exploits data redundancy to compensate node failures (Solis and Obraczka, 2003; Sugihara and Chien, 2005). Depending on the accuracy bounds required for a specific application, a node may need to communicate some of its information to the sink that is incorporated in the model so that the accuracy bounds are met.

Another characteristic of sampled sense data is the distribution of sampled source data. The distribution of data is usually specified in terms of location and pattern (Li et al., 2005). The location is specified in terms of Cartesian coordinates (x, y, z) , while the pattern falls into one of two categories: regular, when data is gathered from sensing nodes which are deployed on a grid; irregular, when sensing nodes are deployed randomly. Many sensor network data belongs to the irregular patterns category. It is sometimes necessary to know the data density besides their distribution. The network density is defined as the number of nodes per unit area.

4.3 The Challenging of Visualisation of Sense Data

Visualisation of sensor network data is a challenging task. Sensor networks are often deployed randomly in unknown environments with some knowledge of the observed phenomena. The large amount of data collected from deployed sensor networks arrives in a *bursty* manner. This makes it difficult to process all the data in a timely fashion so that it can be used as an input to visualisation systems (Pai, 2007). Furthermore, most scientific visualisation techniques require data to include connectivity information, which is not provided by a scattered data set. Hence, highly efficient visualisation schemes operating directly on raw scattered data are necessary (Hamann and Joy, 2007).

Furthermore, in wireless sensor networks, it is increasingly becoming important to have a live picture of the changing environmental variables. This can be achieved if data about the phenomenon is sampled at a suitable rate. In live data representation applications, where a client is interested in the current picture of the environment, it is important to collect readings from the network at a rate as close as possible to the rate of change of the monitored environmental variable. In other applications, such as temperature and pressure monitoring, when the measured phenomena changes linearly over time and space, the sampling rate can be reduced so that node resources are used effectively.

Practically, a point measured on a surface represents the environmental conditions over an area of a certain size; hence it should be possible to generate a complete view of the observation field of a wireless sensor network using a discrete set of observation points. The sampling interval over an unknown surface is defined depending on the desired degree of accuracy and fidelity. The problem is to define how to adequately represent the observed phenomena by a limited number of elevation points, that is, what sampling

interval to use with an unknown surface? To select between different sampling strategies when data is acquired from a sensor network many factors must be considered including: application, level of accuracy, the nature of the sampled data, the nature of the monitored environment, and node distribution and density. Data acquisition strategy is of high importance in sensor networks and is directly related to routing and single node capabilities. When sampled data is used to visualise a certain observed phenomena results are usually good as the sample (Li et al., 2005).

Finally, in large-scale wireless sensor networks it is a non-trivial task to visualise observed phenomena given the problems of sparse, inaccurate, high density, and irregularly distributed data, in addition to the limited physical resources. Therefore, we aim to define a method suitable for real-time visualisation and analysis suitable for sensor network data. This includes the specification of a modular scattered data interpolation package for implementing suitable interactive viewers for time-varying data.

4.4 Benefits of Sense Data Visualisation

Data visualisation in wireless sensor networks has the ability to bridge the gap between the physical and logical worlds, by using the gathered information from the physical world and communicating that information to the end-user in a compact and often easy to understand way. Data visualisation helps to deal with this flood of information, integrating the human in the data analysis process. The main advantages of the application of visualisation techniques in wireless sensor networks are:

1. Visualisation can help in managing the huge amount of data coming from a sensor network. Visual data exploration can easily deal with large, highly non-homogeneous and noisy amounts of data.
2. Maximisation of useful information return. Visualisation is a funda-

mental tool for the communication of information in a compact and easy to understand way. It allows the user to gain insight into the data, draw conclusions and directly interact with it. Visualisation not only helps to answer questions that the user has, but it elicits questions that he did not even think of before.

3. Interactive visualisations benefit from dynamic queries which are a valuable tool to explore data.
4. More reliable information than is possible from individual sources. Although individual devices have limited resources, the true value of the sensor network systems comes from the emergent behaviour that arises when data from many places in the system is combined into a meaningful presentation (He et al., 2004; Becker et al., 1995). The bandwidth of data transferred in a picture is much bigger than having a human look at log files or textual data.
5. Detection of higher-order relationships between different sensors. Relationships become apparent. Sometimes they are completely hidden without visualisation.
6. More efficient data and information representation. Visualisation reduces analysis and response times. Going through thousands of lines of data points is definitely slower than looking at a few graphs of the same data. It is a valuable tool to communicate information in a compact and often easy to understand way.
7. Visual data examination does not require deep understanding of complex mathematical or statistical algorithms. Visualisation techniques provide a qualitative overview useful for further quantitative analysis (Becker et al., 1995). It definitely reduces analysis and response times.

4.5 Sense Data Visualisation: Maps

With the increase in applications for wireless sensor networks, data manipulation and representation have become a crucial component of sensor networks. Approaches to processing and interpreting the data gathered by a sensor network are an urgent demand.

The integration of data visualisation tools and the raw data sent by the sensor network makes the sensor network system useful to different potential users. Visual formats, such as maps, can be easily understood by people possibly from different communities, thus allowing them to derive conclusions based on substantial understanding of the available data. Maps are effective to understand the spatial distribution of environmental features since humans can use their natural interpretation capabilities to understand colours, patterns, and spatial relevance. A map is a visual representation of an area, although most commonly used to depict geography, maps may represent any space, real or imagined, without regard to context or scale such as weather data mapping (Buisseret, 1992). However, a map could be overlaid over a geographic map to enable observation of the data in a real-world map.

In a sensor network, a map may be used as an information representation and extraction tool in which visual features such as symbols and colours are used to code different attributes of the data to provide the information for end users to analyse and examine. These unique visualisation and analysis benefits offered by maps make them more visually communicative, they imply the distributions and states; provide information about spatial patterns; and imply the association of diverse phenomena.

The human interpretation capabilities suggest the importance of expression methods such as how to represent spatial data on a map. Maps can be either static or dynamic and allow data representation on two-dimensional or three-dimensional space. They allow the user to infer the actual sizes and distance between objects. The users can zoom in or zoom out respec-

tively meaning showing more or less details. Furthermore, maps allow the extraction of information that can not be obtained by looking at sensor readings separately and are more efficient to compute in both time and energy. For instance, maps may capture trends or correlations among sense data. Where there is no operating sensor, predictions can be made using these spatial and temporal correlations among sensor readings. Finally, a map provides a higher-level information-rich representation which was found suitable for informing other network services and the delivery of field information visualisation. This information-rich representation satisfies the various requirements for the sensor network system end users.

4.6 Chapter Summary

This chapter has discussed the characteristics of data collected from a sensor network as well as the benefits and challenges in visualising such data. It is proposed that a map format is a suitable visualisation and data extraction tool of dynamic sense data. The map was found to be an intuitive and easy to understand visual format. Also it provides a visual interface for targeting queries for generating detailed maps from a subset of the sensors in the network. The understanding of sense data acquired through maps fulfils the ultimate goal of sensor network deployments which is not only to gather the data from spatially distributed sensor nodes, but also convey and translate the data for scientists to analyse and study.

Chapter 5

A Mapping Service for Wireless Sensor Networks

Wireless sensor networks typically gather data at a discrete number of locations. However, it is desirable to be able to design applications and reason about the data in more abstract forms than points of data. A data extraction, processing, and visualisation mapping service is one way in which this can be done. This chapter explores an implementation of this mapping service and its merits and shortcomings are discussed. The service should be based on in-network processing and would be applied to flat, computationally homogeneous networks. Additionally, an initial application of the service in the form of isopleth generation is presented. Finally, the improvements required to create more sophisticated applications and services are discussed and the benefits these improvements would bring are examined.

5.1 Introduction

Wireless sensor networks based monitoring of real world phenomena often requires high-density, high resolution sampling in order to provide a useful insight into the phenomena (Estrin, 2007). Consequently, such wireless

sensor networks are likely, and indeed expected, to generate vast streams of data; gathering the data and managing it effectively (energy wise and computationally) represents a considerable challenge. At the high application level, the users of deployed wireless sensor networks are less interested in the actual sensor readings than they are in the collective information content of the data sets. In this line, research efforts have led to developments such as TinyDB (Madden et al., 2005) for retrieval of data, which uses the abstraction of Structured Query Language (SQL) to effectively hide the details of data collection, buffering, and transmission, and separates these low level concerns from the application and hence the user. However, the network type for which the abstractions have been developed for are centralised, sink based systems, whereby the information extraction happens at a remote, computationally powerful terminal. Moreover, there are severe limits to the existing data retrieval mechanisms, including issues of temporal requests and locality of network data. The benefits of abstractions such as this, nevertheless, have been expounded in the literature (Meng et al., 2006; Wood et al., 2003; Hellerstein et al., 2003) and provide a strong motivation for the concepts developed in this work.

In this chapter, a service is proposed which allows the production of maps of an arbitrary level of detail upon requests injected into the network by the user, or pre-programmed as responses to network events.

5.2 Understanding the Problem of Map Generation from Sparse Data

Map generation is essentially a problem of interpolation from sparse and irregular points. Given a set of known data points representing the nodes' perception of a given measurable parameter of the phenomenon, what is the most likely complete and continuous map of that parameter? In the field of computer graphics, this problem is known as an unorganised points

problem, or a cloud of points problem. That is, since the position of the points in xy is assumed to be known, the third parameter can be thought of as height and surface reconstruction algorithms can be applied.

Simple algorithms use the point cloud as vertices in the reconstructed surface. These are not difficult to calculate, but can be inefficient if the point cloud is not evenly distributed, or is dense in areas of little geometric variation.

Approximation, or iterative fitting algorithms define a new surface that is iteratively shaped to fit the point cloud. Although approximation algorithms can be more complex, the positions of vertices are not bound to the positions of points from the cloud. For applications in wireless sensor networks, this means that we can define a mesh density different to the number of sensor nodes, and produce a mesh that makes more efficient use of the vertices. Self organising maps are one of the algorithms that can be used for surface reconstruction (Yu, 1999). This method uses a fixed number of vertices that move towards the known data.

Note that surface reconstruction on typical non-overlapping terrains is equivalent to sparse-data interpolation. This kind of geometric parameter interpolation has been shown to work well for reconstructing underlying geography when the entire network has been queried (Shuttleworth et al., 2006). However, It does not extend well to variable surfaces or overlapping local mapping, since it requires a complete data set to define the surface.

A more general method is interpolation by inverse distance and, specifically, Shepard interpolation (Shepard, 1968) which improves on it. The simple inverse distance algorithm, used in wireless sensor network applications before (Tynan et al., 2005), is discussed in details in Section 7.1.

Shepard devised a number of improvements to this basic algorithm to limit the effect of distant points, make use of the direction of the relationship between known points and the point to be interpolated and to incorporate information on the slope between known points. The simple algorithm has

been implemented and used by the mapping service to interpolate between sensor readings.

5.3 A Framework for Mapping Sense Data: Centralised Mapping Service

In this section, our first steps towards a general data extraction and visualisation mapping service are presented. We define a simple service, discuss our simulation of networks that implement it and present an example application built on the service.

Centralised Mapping Service Details

The purpose of any wireless sensor network is to gather data. In the simplest systems, the collected data might be instantly reported. In more complex systems, the data will be processed, aggregated, distilled and acted upon within the network, possibly with only selected data being reported back to a monitoring user. The wireless sensor nodes are usually more than capable of performing these tasks, as long as the application developer creates the required software, constructing specific functions from what has become increasingly generic hardware.

There has been a consistent effort to change the mechanism of use of certain capabilities in wireless sensor network, to simplify and abstract them, turning them into services of the network rather than being the result of coordinating the services of individual nodes. TinyDB's retrieval of data (Hellerstein et al., 2003), for example, uses the abstraction of SQL to effectively hide the details of data collection, buffering, and transmission.

The primary purpose of wireless sensor network is to collect and transmit data, but other capabilities have arisen to support this goal. Just as clustering, routing and aggregation allow for more sophisticated and efficient

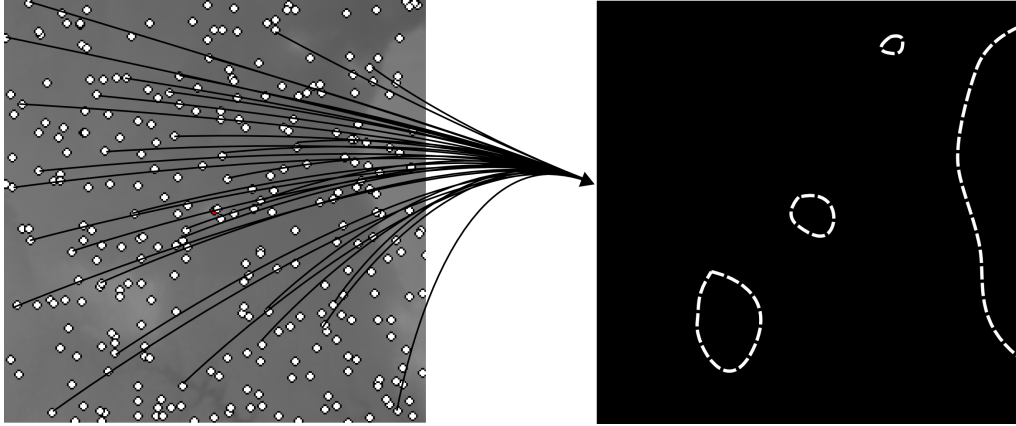


Figure 5.1: Data is collected at a central point and a global map is computed at once.

use of the network resources, a data extraction and visualisation mapping service would support other network services and make many more applications possible with little extra effort. Following on from such work and moving to a slightly higher level of abstraction, we propose a new network service: *map generation*. This map generation service is studied in detail in Chapter 7.

In the simple centralised data extraction and visualisation mapping service, all dumb points of data are transmitted to the sink and a global map is computed at once. Figure 5.1 shows an example of data collection at a central node out side of the network. This approach will produce high quality maps as the entire network data is used for building the map. However, a fully centralised data collection is not always feasible. Data needs to travel to a central point which possibly causes bandwidth bottlenecks especially around the sink. This also may cause large drains on resources. Furthermore, a big portion of this return data is not useful, particularly the unchanged modalities and redundant packets. Also, this method usually does not scale with the network size and requires high communication overhead to maintain up-to-date maps. Finally, in some scenarios, it is desirable

or necessary to process this information on site and, as a result, in-network mapping provides a critical solution to in-field data analysis.

The service is made up of three core components: routing, interpolation, and application. The routing component encapsulates the communication architecture used for data delivery to the sink. In this work, the MuMHR routing algorithm (see Chapter 6) was used for data communication. The interpolation component encapsulates the new network service, map generation. Interpolation is most often used to estimate an unknown value between known values by utilising a common mathematical relation. This process makes it easy to predict values at points where there is no sensing device. This is done by using other known sensed values that are located in sequence with the unknown value. Finally, the application module contains the entire code specific to the user application functionality. An example application is isopleth, navigation maps, etc. Figure 5.2 shows the architecture of the proposed mapping service.

Benefits of a Distributed Mapping Service

We predict many benefits will result from the development of an efficient and flexible distributed mapping service.

Networks of nodes are built to solve problems. Many problems are essentially problems of interpolation between points and it is this set of problems that this work intends to address with a mapping service. Defining lines of constant height or pressure, contours and isobars, for example, requires knowledge not just of measurements at a few scattered locations, but also the likely values between them. Contours and isobars are, in fact, specific types of isopleth and a simple early implementation of isopleth determination is presented later in this chapter.

Another problem, that is at heart simply a problem of interpolation, is surface reconstruction. While mesh-based models are feasible and useful for specific applications in the context of wireless sensor networks (Shuttle-

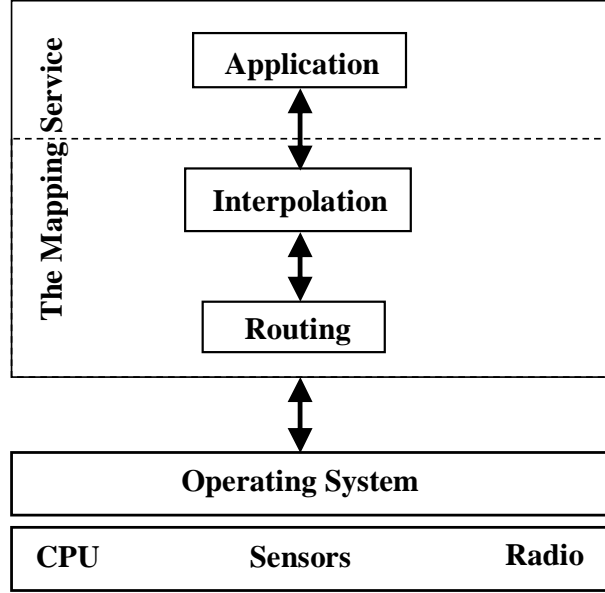


Figure 5.2: Architecture of the centralised mapping service.

worth et al., 2006), an interpolation-based model of surface reconstruction offers benefits such as being locally applicable, of arbitrary detail and easily subjected to image processing techniques.

Other aspects of wireless sensor networks that would benefit from such a service include clustering, in which geographical context might be useful; deployment, for which the mapping service could provide information on network density related to terrain or phenomenon complexity, and so on. As well as these easy to identify benefits, it is likely that having such a service would make new applications obvious, just as there has been a recent increase in new Internet applications combining existing services to produce new and until now unthought-of “mashups”.

Indeed, the efficacy and efficiency of applications based on such a service

are tied to how well the service is implemented. The implementation of the mapping service presented in this section, and the example application built upon it, are not optimal. In short, they are inefficient and naive. However, while more sophisticated implementations of the algorithm are being devised and tested, this simple approach gives us an opportunity to investigate the usefulness and validity of its application.

Experimental Setup for Mapping Applications

In the following experiments, the SenSorPlus wireless sensor networks simulator (see Chapter 2) was used. A section of the Grand Canyon height map (McCabe, 1998) is used to study the performance of the centralised mapping service. Random graphs were dispersed between $(x = 0, y = 0)$ and $(x = 256, y = 256)$ with the sink at location $(x = 128, y = 128)$. The transmission range of each node is bound to $250m$. The processing delay for transmitting a message is randomly chosen between 0 and $5ms$, simulating real-world characteristics of low-power radio transmission. We assume an ideal MAC protocol within which no collisions can occur. This is the same approach adopted in all prior work (Dai and Wu, 2004; Lou and Wu, 2003) to compare the efficiency of heuristics.

Using this network configuration, data was gathered to interpolate the landscape and draw the isopleth from the interpolated output. The landscape image is fed to the simulator and sensors will “perceive” the colour intensity at the (x, y) position of the landscape image corresponding to their coordinates as gathered sense data. After cluster heads are selected and paths are established, an external observer can choose to query any node to collect the information necessary to interpolate the surface from the network. Every node responds to the query message by sending its sense value and position through the cluster head. This collected information could then be used to build the network surface using the Shepard interpolation algorithm.

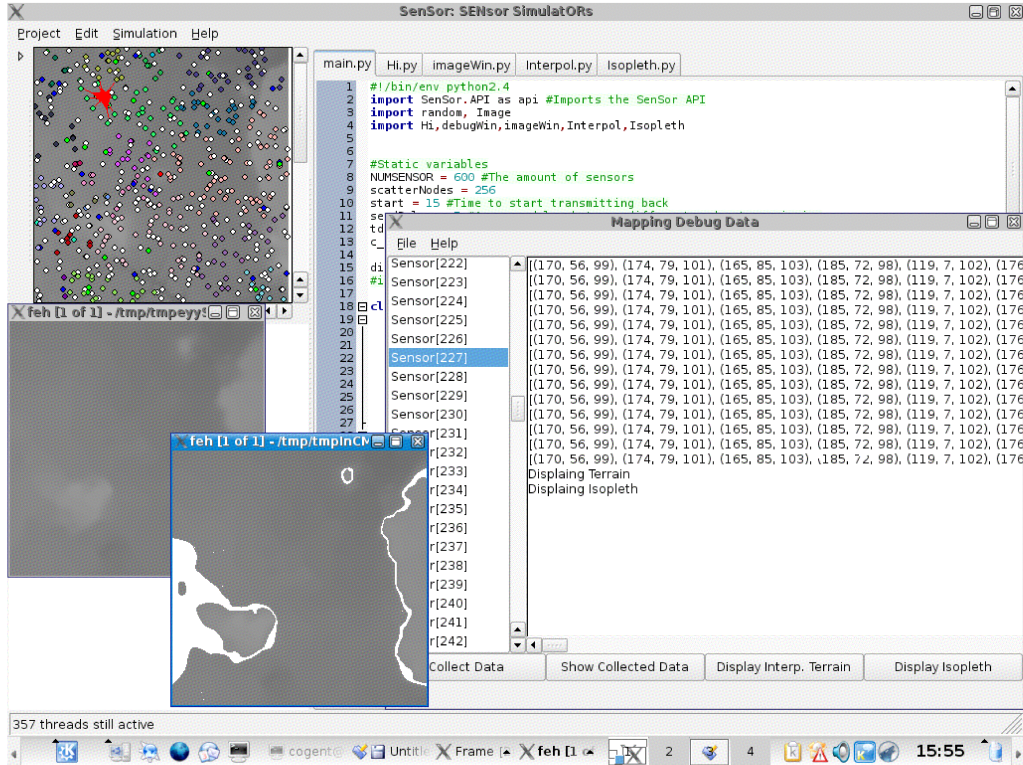


Figure 5.3: The mapping extension of SenSorPlus, the wireless sensor network simulator. The figure shows the sensor nodes, communication between nodes, querying interface, input data, generated map, and an isopleth application on top.

Figure 5.3 shows a screenshot of SenSorPlus running a simulation of 600 nodes randomly scattered over a height map of a section of the Grand Canyon, taken from data recorded by the US Geological Survey.

Experimental Isopleth Generation

The determination of isopleth or contours is useful in a number of applications and the ability to find the “edges” of a phenomenon has been put forward before as a useful operation (Elmusrati et al., 2005). Such an abil-



Figure 5.4: Interpolated field generated from 600 simulated nodes randomly distributed on the surface from 2D map in Figure 7.3

ity would enable the systematic finding and delineation of the borders of phenomena such as gaseous emissions or freezing conditions, creating height contour maps, calculating lines of sight, and so on.

To highlight this potential use of a mapping service, an implementation of isopleth generation is presented. This is a simple implementation, with little regard for efficiency, but nevertheless provides a genuinely useful output for the purposes of visualisation or phenomenon edge delineation. With no optimisation or refinement to reduce communication costs, the algorithm is simple. A request is made to any node, giving the value of the required isopleth. A height of 120 metres, for example, if contours of height were needed. This node is then responsible for collecting the information needed to produce the result. A threshold figure is also given, t , so that values $\pm t$ are included.

The simple algorithm presented here causes this node to then query every other node in the network for their value of the parameter in question. Once all data is collected, the interpolation is performed, and every value

matching the required isopleth is recorded as being part of that isopleth.

This algorithm has been implemented using the simulation software SensorPlus and the simple API described in Section 5.3. Figure 7.3 shows a section of the Grand Canyon height map that we use in our experiments. If the isopleth generation algorithm is applied to this data directly, the result will be such as that shown in Figure 5.5-(a). This contour was generated for a height of 116 units and a threshold of 1.

To present the result of generating isopleth on interpolated fields as described above, Figure 5.5-(b) shows the result of generating an isopleth with the same parameters as in Figure 5.5-(a), on the interpolated field shown in Figure 5.4.

Clearly the interpolated terrain is similar to the real surface. Determining exactly how close the similarity is, and what the algorithmic limits to the accuracy of the representation, is a problem to be investigated in Chapter 7 and Chapter 8. Consider, though, that the information used to reconstruct the surface in Figures 5.4 and 5.5-(b) is just 600 points, while the original is recorded as 65,536 points. Taking the position of the nodes into account as extra information, the reconstruction is built using less than 1% of the original data.

In a more advanced implementation, this algorithm would be replaced by one that begins with an efficient search for the first matching value and then a process of extending the search along the isopleth as it is discovered.

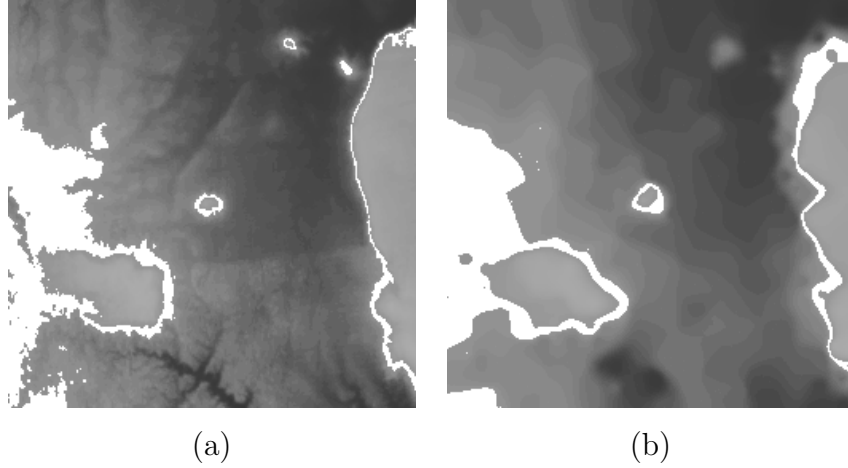


Figure 5.5: (a) A contour at 116 units drawn over 2D map in Figure 7.3. (b) A contour at 116 units drawn on the interpolated field in Figure 5.4

Discussion

Leading directly from the work in this chapter, there are a number of avenues that need to be followed.

The centralised approach above collects all of the data in a central location and processes it there. In networks of a few nodes, this might even be the most sensible solution. However, in networks of nodes large enough to be useful, the number of transmissions needed to accomplish the task would become so large as to require a more intelligent solution. The cost of communication in this initial system is high because the interpolation step, the mapping service, is fairly simple and requires that all data is collected to a point. For the simplest case, in which the node collecting data is in direct communication with all other nodes in the network, the number of hops is simply $N - 1$, for N nodes. However, this is an unlikely situation in any real world application. The likelihood is exponential growth with N .

Although the cost, in terms of power of transmission of data in wireless sensor networks far outweighs the usual processing load, the nodes are

most often power-limited devices with, at best, a fixed schedule of battery replacement, and so computational drain on this resource cannot be completely overlooked. For small collections of data of around 50 nodes interpolating 65,536 points (a square field (256×256) takes very little time, just a few seconds, on a desktop computer and would take just a little more on modern sensor nodes. In a real deployment, the number of sensor nodes might easily exceed 1,000, depending upon the required density, drastically increasing computational expense. The acceptability of such a computational load in a wireless sensor network is debatable, but in this case would probably be unacceptable for most deployments.

The simplest improvement is to develop a complete implementation of Shepard's refinements and assess how this affects the accuracy of the interpolated field when compared to a known real surface or phenomenon. Although the algorithm presented by Shepard is decades old, it is often only implemented in its most simple form and it would be of great benefit to have empirical evidence of the performance of the various improvements suggested by Shepard, when applied to data likely to be perceived by a sensor network. A particularly interesting feature of the refinements is that they deal with limiting the data included in the weighted average, or adjusting weights based on context. These improvements might be instantly applicable as ways to reduce communication costs as well as improve the quality of the interpolation. That is, if local information is required, then only local nodes need be queried.

Taking this idea further, we arrive at the idea of a mapping service that allows for local querying. The proposed solution, currently being developed, is an application of the more advanced mapping service outlined above.

An isopleth for a given parameter at a given value can be found by a step-by-step search for the next point in the map at the same level. That is, a single point is used as the start of the isopleth and the next is found by searching the neighbourhood in progressively more detail. The process is re-

peated until the end of the isopleth moves out of the mapped region or meets the beginning. The highest level of detail that is searched will determine how fine the isopleth is. The same improvements intended to decrease the cost of transmission also reduce the cost of computation. The change from global to local area interpolation reduces the number of transmissions and makes the task of interpolation significantly smaller. Additionally, since the local interpolation could be performed by local nodes, the computational cost is not just reduced, but also distributed. That is, rather than individual nodes taking big losses in battery life and becoming unusable quickly, the entire network's batteries degrade at a more constant rate. In chapter 9 we investigate the application of the mapping service to other network exposures such as barriers that might cause discontinuities in the interpolated map, contours, or many other sophisticated applications.

5.4 Chapter Summary

In this chapter we have presented our very early research into mapping and the applications of mapping in wireless sensor network. The results indicate that this area of research is pertinent to modern wireless sensor network, and in this section initial steps towards exploring it have been taken. Applications have been identified, such as isopleth generation, that the service would make simple to develop, and challenges that need to be met before such a service is feasible. Challenges such as local interpolation, required to make the service efficient enough to be deployed. This simple centralised mapping service is the exposition of the start of a fully distributed mapping service. The problems and limitations described above are the opportunities intended to be taken and the lines intended to be followed in the design of the distributed data extraction and visualisation mapping service.

Chapter 6

A Communication Paradigm for the Mapping Service

The definition of the mapping service given in Chapter 5 changes the mechanism of use of certain capabilities in wireless sensor networks. In this work the routing and the interpolation capabilities of the network were chosen to demonstrate how we can turn these capabilities into services of the network. The detailed study of wireless sensor networks communication paradigms provides insights into how the mapping service may exploit the routing capability of the network to solve the global network data mapping problem through localised and distributed computation. Moreover, this study gives solid and thorough understanding of various communication related issues, such as timeliness and energy efficiency that have deep influence on the performance of the mapping service.

In this chapter a self-organising, cluster based routing paradigm called *Multi-path, Multi-hop Hierarchical Routing* (MuMHR) is proposed. This routing paradigm deals with a large number of mapping requirements identified in Section 2.3. MuMHR performance is evaluated via simulation using NS-2 and Dingo. Simulation results show that MuMHR performs better than LEACH, which is the most promising hierarchical routing algorithm

to date; MuMHR reduces the total number of set-up messages by up to 65% and enhances the data delivery ratio by up to 0.98.

6.1 Introduction

The stream nature of the transmitted data, the limited resources, and the distributed nature of sensor networks bring new challenges for the data extraction techniques that need to be addressed. In such networks, the data from the entire network can be extracted after a certain period of sensing and local storage. However, there is a trade-off between energy consumed and the data extracted in wireless sensor networks. Since communication is often the most expensive operation for a sensor node, the mapping services underlying data communication paradigm must be energy efficient.

In wireless sensor networks routing is application dependent and design goals vary among different applications. For instance, many applications require real-time communication. For example, a fire fighter may rely on timely temperature updates to remain aware of the current fire conditions. Therefore, wireless sensor networks must meet the delay requirements at minimum energy cost. Hence, routing mechanisms have considered the characteristics of sensor nodes along with the application and architectural requirements.

In section 2.3 and Appendix B some of the most important hierarchical routing protocols were reviewed. Although the performance of these protocols is promising in terms of energy efficiency, most of them come with no guarantee of Quality of Service (QoS) required by real-time and communication demanding applications. As discussed in the literature review (section 2.3), we need a protocol that has the following characteristics: energy efficiency, scalability, message loss robustness, and provision of timeliness.

To address these challenges, a self-organising cluster-based routing protocol called *Multi-path, Multi-hop Hierarchical Routing* (MuMHR) is pro-

posed. MuMHR has the same underlying benefits as LEACH (Heinzelman et al., 2000) and its derivative protocols but provides for multi-hop communication, and increases robustness by using multiple communication paths. Also, in comparison to LEACH and most derivatives, this protocol reduces the number of setup messages, and thus extends the network life. With MuMHR, robustness is achieved by each node learning multiple paths to its cluster head and by the election of cluster head backup node(s). Energy expenditure is reduced by shortening the distance between the node and its cluster head as well as by reducing the setup communication overhead. This is done through incorporating a number-of-hops metric in addition to the back-off waiting time.

6.2 MuMHR Design Objectives

In wireless sensor networks, many of the communication protocols are developed to meet application-specific design goals and requirements. Therefore, the design objectives of routing protocols vary according to these requirements. Fundamental design objectives of sensor networks include reliability, accuracy, flexibility, scalability, energy efficiency, fault tolerance, self configuration, timeliness, etc. Most of the existing routing algorithm focus on one or two of these design objectives and ignore the others. For example, Wang et al. (2006) presented a Hierarchical Multiple-Choice Routing Path Protocol (HMRP) with energy efficiency is the key design factor. Also, ARPEES (Tran Quang and Miyoshi, 2008) is an adaptive multi-hop routing protocol for wireless sensor networks that aims on spreading energy consumption among sensor nodes to prolong the lifetime of the whole network. TVHRP (Wei et al., 2008) is another hierarchical routing protocol based on the shortest path to handle dynamic topological variation efficiently. In TVHRP each node communicates with the nearest nodes in its upper and lower layer. The nodes located in the layer nearest to the topological

variation layer are first triggered to respond and optimise the network performance. Chun-yan et al. (2008) proposed a new clustering hierarchy tree routing algorithm based on LEACH to ensure that cluster heads are uniformly distributed in the network. Unlike previous work aiming to maximise energy efficiency and network lifetime, Yang et al. (2006) proposed a routing scheme, called Minimum Energy Spanning Tree for Efficient Routing (MESTER), which aims to maintain an accurate and timely data collection for as long as possible. Kumar et al. (2009) assume that a percentage of the population of sensor nodes is equipped with the additional energy resources and they studied the impact of heterogeneity of nodes in terms of their energy in hierarchically clustered sensor networks. Lee et al. (2006) have considered an event detection heterogeneous network with two types of nodes, type-1 and type-0 node, with type-1 node having more battery energy than type-0 node. A key design implication is that the energy supply built into the nodes and that the heterogeneous node is line powered, or its battery is replaceable. Recently, Kim et al. (2008) introduced a cluster head election method using fuzzy logic to overcome some of the defects of LEACH. They investigated that the network lifetime can be prolonged by using fuzzy variables in homogeneous network system. Tubaishat et al. (2004) proposed an energy-efficient multi-hop hierarchical routing protocol. The primary goal of this protocol is to provide secure data communication. This goal was achieved, however, at the expense of energy consumption. Jawhar et al. (2009) presents and evaluates a protocol for Linear Structure wireless sensor networks which uses a hierarchical addressing scheme designed for the special issues and characteristics that are specifically related to this kind of network.

MuMHR aims to combine many of the important design objectives which are requirements for most wireless sensor network applications. It attempts to achieve an acceptable level of energy efficiency, timeliness, scalability, fault tolerance, adaptation to topological variations, load balancing, uni-

form cluster formation, and reliability at the same time. Many protocols out perform MuMHR in one of these design objectives, e.g. MESTER out performs MuMHR in timeliness, however, MuMHR can achieve comparable results and most importantly a more balanced overall system performance at a much lower complexity. MuMHR is designed to be a general routing protocol that can be customised to suit different application requirements. For example, in time critical applications, MuMHR can be modified to assign priority for various communicated packets.

Chapter 8 explains how the MuMHR routing protocol links to the mapping process and how the mapping function is informed by the routing algorithm.

6.3 MuMHR Algorithm Details

In this section, the properties of the proposed routing protocol, MuMHR, are described. The main objectives of MuMHR are to provide substantially energy-efficient, timely, and robust communication.

The energy efficiency is achieved by load balancing at two levels: (1): Network level, through traffic multiplexing over multiple paths; (2) Cluster level, by rotation of the cluster heads every given interval of time. This prevents energy depletion resulting from constantly using the same path for transmission or particular nodes being cluster heads for a long duration.

The MuMHR multi-path property is not designed simply for load balancing but also for when path failures occur. When a path fails, an alternative path can be immediately used which allows the protocol to dynamically adapt to failures without delays or degradation in the quality of service. At the cluster set-up time, one or more nodes are chosen as cluster head backup node(s). Backup cluster head nodes substitute for the cluster head in some types of failure or when the current cluster head decides to reduce its participation in the protocol if its energy level approaches a certain

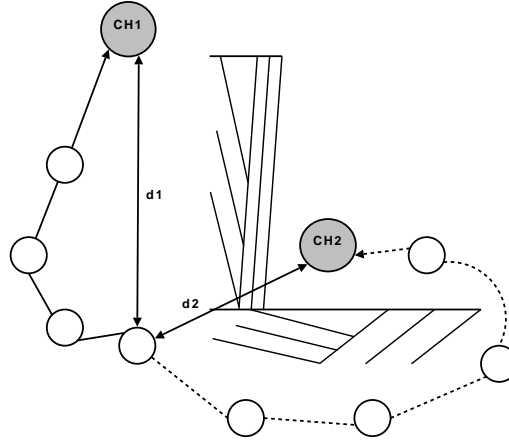


Figure 6.1: Routing adaptation to obstacles using the number-of-hops. The filled nodes are cluster heads. Horizontal and vertical bars forming L shape are obstacles. The solid lines are path from node $n1$ to CH-1 and the dashed lines are path from node $n1$ to CH-2.

threshold. For instance, if the current cluster head decides to hand its role to the backup node, it notifies the respective backup node and forwards to it necessary information, such as the backup nodes list, to avoid a complete cluster set-up phase.

Multi-hop communication is an essential property of any scalable wireless sensor network because sensor nodes transmit using a limited radio power. Each node only communicates with a limited number of neighbours located within its radio range, regardless of the network size. When two nodes that are out of each others radio range need to communicate, they rely on intermediate nodes to establish a connection between them. Multi-hop communication not only saves the nodes the radio transmission power, it also has the following benefits (Gui, 2005).

1. Spatial reuse: simultaneous transmission and receiving sessions are possible for node pairs that are out of range of each other
2. Obstacle "conciliation": a multi-hop path can go around an obstacle

that obstructs direct radio connection between two nodes as shown in Figure 6.1

3. Distributed processing: a multi-hop topology allows the local processing of sensor data among nearby nodes, and only delivers the processed information to the remote user station

On the other hand, multi-hop communication has also introduced significant complexity to the routing algorithms. In MuMHR, the *number-of-hops* metric was introduced. This metric measures how far the sensing node is from the cluster head. As the number-of-hops in a particular path increases, the power consumption due to the higher number of transmissions increases. Therefore, shortening the route will result in significant performance gain over other multi-hop routing protocols. The number-of-hops metric is used to:

1. Select the nearest cluster head node, which saves energy and reduces messaging needed to bridge the distance between the cluster head and the sensor node
2. Allow a node to find the shortest path to the selected cluster head

Moreover, MuMHR implements a metric called the *back-off waiting time* to decrease the total set-up traffic and aid the formation of more geographically uniform clusters. During the back-off waiting time, sensor nodes accumulate received cluster *advertisement* messages and only consider the message with the smallest number-of-hops received during that time. This aids blocking advertisement flooding from unnecessarily reaching neighbouring clusters. Both the number-of-hops metric and the back-off waiting time allow more energy efficient cluster formation.

Many of the proposed protocols in the field of sensor networks show poor fault-tolerance in the face of frequent path failures (Liu and Seah, 2004). MuMHR provides fault tolerance through a multi-path routing strategy.

The multiple paths are built by nodes during the set-up phase through received identical messages from different neighbouring nodes. For example, multiple paths to the cluster head are built from received multiple copies of an identical advertisement message from different neighbouring sensing nodes. Each node joins the nearest cluster for a certain period of time, t_c , before it re-registers with that cluster head again or registers with a new one. Let t_r be the time remaining for a cluster head to start a new cluster formation round. A cluster head starts a new cluster formation round by handing the cluster head role to the backup node. In this approach, the worst time, t_w , to recover from a path failure is the time taken for a node to renew registration with a cluster head. This is written as: $t_w = t_c - t_r$. If all paths that a sensor node has learned fail, the node will broadcast its data to its neighbours. Neighbours will then pass the message to their cluster head. The cluster head may receive multiple copies of the same message and eliminates redundancy by applying aggregation.

The operation of MuMHR can be split into two phases: the setup phase and the data transmission phase.

Set-up Phase

During the set-up phase, cluster heads are selected and clusters are created. We assume that a simple addressing scheme is in place. The sink randomly selects 5% of the nodes as cluster heads and floods the network with this information. Every node that receives the sink *discovery* message changes its state from *waiting* to *discovered* and examines the message to check whether it has been selected as cluster head or not. If yes, it starts a new cluster by broadcasting an *advertisement* message. Otherwise, it forwards the discovery message to its neighbours. Every node will remember the node from which it has received the discovery message as the immediate neighbour nearest to the sink. This path will be used only in cluster failure situations or when the node is not assigned a transmission time slot in its

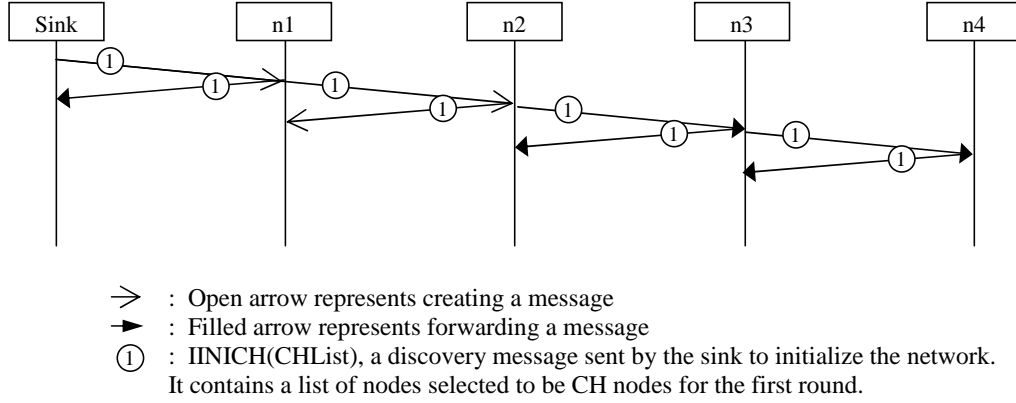
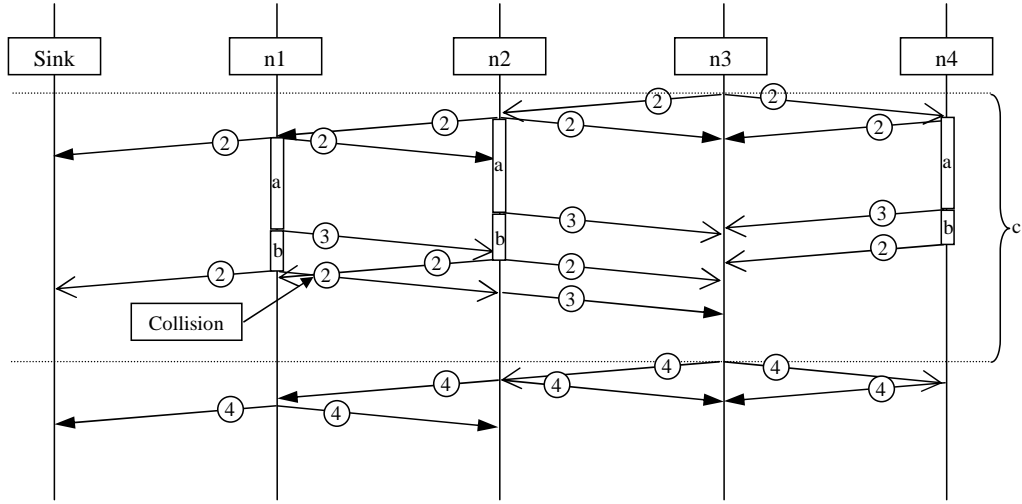


Figure 6.2: The sink broadcasts a discovery message, *INICH*, that contains a list of nodes randomly selected to be cluster heads in the first cluster round of the protocol life. As in flooding, a node hearing two broadcasts will forward only the first received message to prevent circular loops. In this figure, we assume that the network is made of four nodes ($n1$, $n2$, $n3$, $n4$) and a sink node. The sink can reach nodes $n1$ and $n2$ directly.

cluster Time Division Multiple Access (TDMA) data transmission schedule. The sink flooding step can be compared to the seeking of a single route in Dynamic Source Routing (DSR) (Johnson and Maltz, 1996), but every node knows only the next hop not the complete hop-by-hop route to the sink, and no node knows a route to any other node. This sink announcement step is depicted in Figure 6.2.

Every cluster head will create an advertisement message that has the number-of-hops parameter set to 0 and broadcasts it to its neighbours. Upon receiving an advertisement message, a sensor node will do the following:

- If the node already belongs to a cluster, then it ignores the received advertisement message
- If the back-off waiting timer is still valid, then it caches the received packet and waits for other possible advertisements



- \Rightarrow : Open arrow represents creating a message
- \rightarrow : Filled arrow represents forwarding a message
- a : backoff time
- b : random transmission time
- c : time the CH waits before it creates and advertise a TDMA schedule
- ② : ImCH(ch_id, numHop), an advertisement message sent by the CH to declare its new role. It contains the CH ID and the number-of-hops to reach it.
- ③ : Wish(ch_id, wish_value), this is a join request sent by a sensing node to a CH. It contains the respective CH ID and a wish value representing its desire to become a CH in the next round.
- ④ : ADV_SCH(order), an advertisement message sent by the CH to all its member nodes. It contains the TDMA schedule that specifies the order of data transmission for each node.

Figure 6.3: This figure illustrates the cluster formation process in MuMHR. In this scenario, node $n3$ was chosen to be a cluster head for the first round in the protocol life. The cluster head node sends an advertisement message to all neighbours declaring its role. All receiving nodes hold for a period of time a before they forward the message or respond to the sending cluster head. After time a passes, the node chooses the message with the minimum number-of-hops and responds to the sending cluster head. The node first sends a join request message to chosen cluster head and after a random delay time b it forwards the received advertisement message to its neighbours. The cluster head waits for a time c to receive all potential join request messages before it creates and broadcast the TDMA schedule. A point to note is the collision which happened as a result of nodes $n1$ and $n2$ forwarding advertisement messages at the same time.

- If the received message has a better number-of-hops metric than the stored one, the latter is deleted and the former is retained

After the back-off waiting time, the sensor node increments by 1 the number-of-hops parameter in the best packet (the packet with the smallest number-of-hops) it received during the back-off time and then broadcasts it to its neighbours. The node will remember the address of the sender, the cluster head, and the node from which it received the message as the nearest neighbour to the cluster head. If a sensor node receives multiple copies of the same advertisement, it selects the one(s) with the minimum number-of-hops parameter. Then the node uses the available energy to calculate a value, v_c , that represents its desire to become a cluster head in the next cluster set-up round. This value is included in the *join request* message that the node sends back to register with the chosen cluster head. The cluster head will extract the message with highest v_c value(s), and adds its corresponding sender to the cluster head backup list and registers the node as a member of the cluster. Compounding different functions into a single multi-purpose message reduces set-up communication overhead and thus makes the protocol more stable and energy efficient. The cluster formation messaging is illustrated in Figure 6.3. When the cluster round time is over, the current cluster head hands the master role to the first node in the backup nodes list. With a single flood heard by the cluster members only, the new cluster head continues its predecessor's role without the need of further communications. The cluster head role will also be handed to the backup node when a fault occurs in the current cluster head node. However, in the study presented here, a limited set of faults were considered for the “handover” such as internal errors and when the energy level approaches a pre-defined threshold. In the case of faults, such as physical damage or fatal internal errors in the cluster head, the nodes will transmit directly to the sink until a new cluster is formed. The handover of the cluster head role process is described in Figure 6.4.

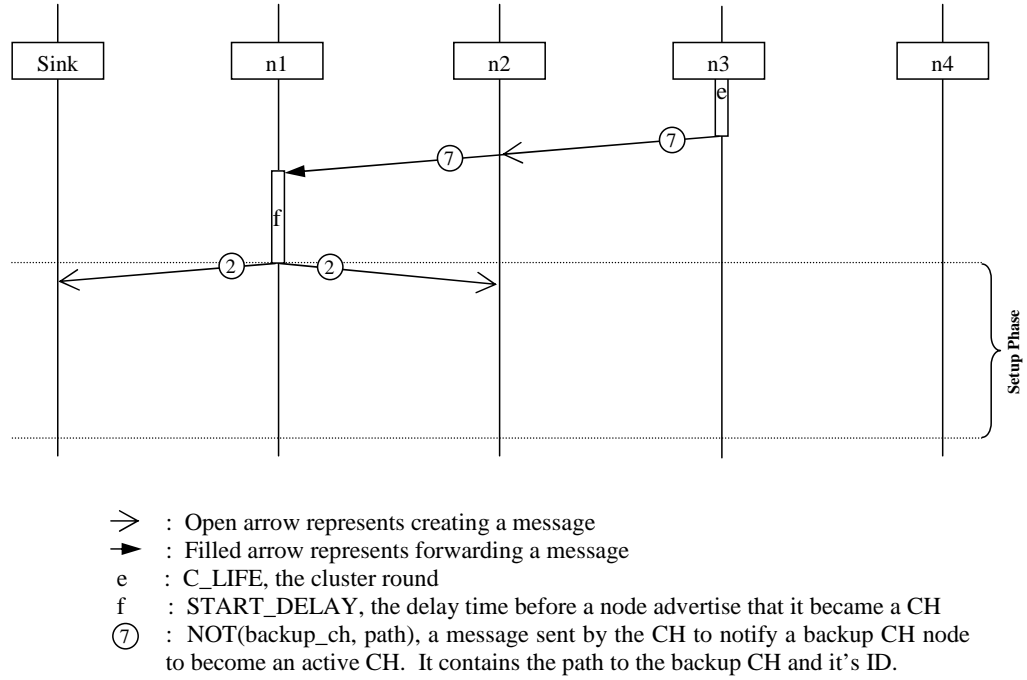


Figure 6.4: This figure illustrates the behaviour of MuMHR when a cluster round ends. The acting cluster head hands its role to the backup cluster head by sending a notification message. The backup cluster head starts a new cluster formation round by sending an advertisement message. In this figure, node $n3$ informs node $n1$ to start a new cluster formation round. After delay f , node $n1$ initialise a new cluster-level setup phase.

Data Transmission Phase

During the data transmission phase, sensing nodes transmit data to their cluster head. The cluster head aggregates the received data before transmission to the sink or immediately multiplexes messages over multiple lines in time critical applications. Each member node transmits data on its assigned time slot scheduled by the TDMA schedule. Furthermore, each cluster communicates with the sink using unique Code Division Multiple Access (CDMA) codes to avoid interference with traffic generated by other clusters. Figure 6.5, illustrates data transmission from nodes to their clus-

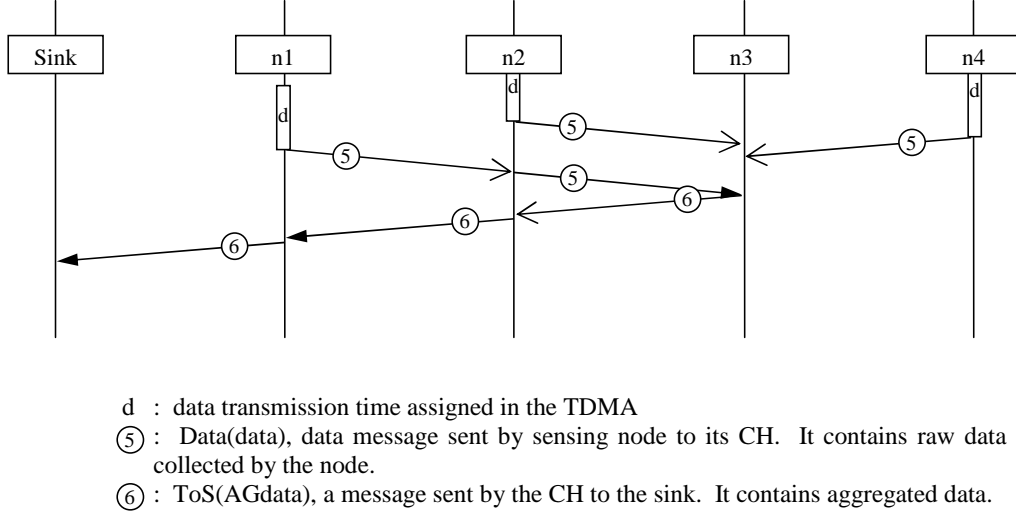


Figure 6.5: This figure illustrates how data routing is handled in MuMHR. In this scenario, there is a single cluster containing three member nodes ($n1$, $n2$, and $n4$) managed by node $n3$. Each node transmits a data message within a time slot assigned by the TDMA schedule. The cluster head $n3$ aggregates all the data messages and send them to the sink when it receives the data from the last node in the TDMA schedule.

ter head. It also illustrates how the aggregated data is sent from the cluster head node to the sink through multi-hop path.

Figure 6.6 represents the whole operation of MuMHR. This figure describes the behaviour of the MuMHR algorithm, particularly, in what order different functions should be carried out and shows how the steps in both protocol phases map to the tasks in the activity diagrams.

Energy Efficient Sleep/Wake Scheduling

Network lifetime maximisation is a key element in the design of MuMHR routing algorithm. Idle sensor node listening consumes a significant amount of energy (Wu et al., 2007). An effective approach to preserve energy is to set the radio to sleep during the idle times and wake it up precisely before

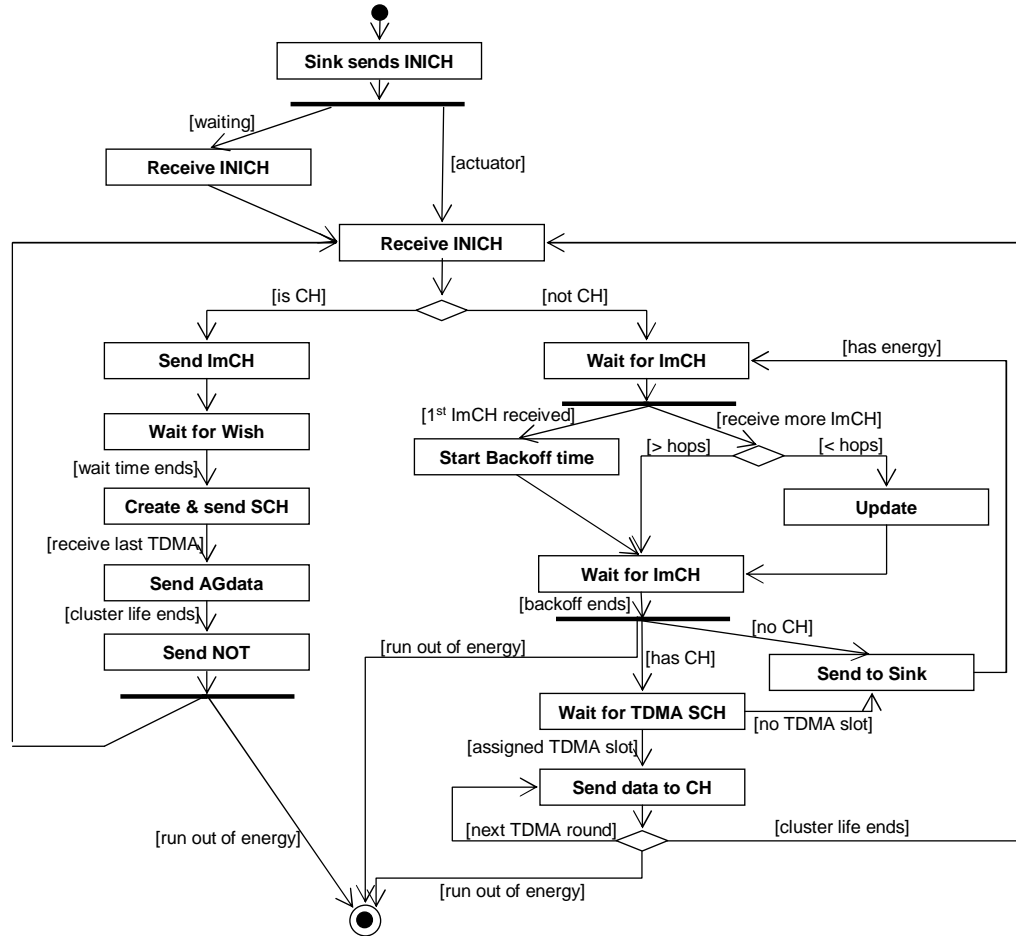


Figure 6.6: UML activity diagram that model the entire work flow of MuMHR algorithm.

message transmission/reception. This is straight forward to implement in single-hop communication networks such as LEACH. However, in networks which involve multi-hop communication it requires precise synchronisation between sending and receiving nodes so that they can wake up simultaneously to communicate with each other. In MuMHR, if a node is in the path of transmission of other nodes it must not sleep before it forwards the messages of its dependent nodes. To sustain a connected graph, sensor nodes sleep times should be appropriately synchronised. In MuMHR, a node n has a set of children nodes N such that for every node, N_i , which belongs to N , n is in the transmission path from that node to its cluster head. Node n goes to sleep only after sending its own data and forwarding all data messages of its children. Sensing nodes make up their children set from forwarding join request messages during the cluster setup phase.

6.4 Optimal Cluster Balancing

In this section we propose an optional cluster balancing plugin called Nutrient-flow-based Distributed Clustering, *NDC*. This plugin can be used with any clustering algorithm. The aims of the NDC algorithm are:

1. To equalise, so far as is possible, the diameter of the clusters
2. To equalise, so far as is possible, the membership of the clusters

The distributed model described here is based around a metaphor of nutrient flow supporting some life-form, such as a mould. The concept is to provide a limited supply of nutrient and allow the nodes to ally themselves with a cluster head which will provide the largest nutrient supply. If properly regulated, this should lead to clusters broadly equalising their membership. In order to minimise the radius of a cluster, it is arranged that some of the supply of nutrient is lost in transit between nodes - the further the distance travelled, the more is lost. It is important that some advantage

be given to nodes that join in a cluster, rather than communicating directly with the sink. For this reason, it is necessary to provide some advantage associated with clustering, as opposed to direct communication. The simplest way to do this is to make the loss of nutrient super-proportional to the distance of a link. Given that in real life, radio propagation obeys an inverse square law, it seems reasonable to make the loss of nutrient proportional to the square of the distance travelled.

Like many distributed route discovery algorithms, this one operates in distinct phases with the network reconfiguring itself from phase to phase. During each phase, nodes try to improve the amount of nutrient available to them. They do this by contacting a local node at random and if clustering with that node will offer a better supply than that which is currently available, then the node changes allegiance. Nodes receiving requests for nutrient from other nodes make an offer back to that node, giving an estimate of the nutrient that would have been available had that node been a member of its cluster. The estimate depends both on the amount of nutrient available, and the number of nodes dependent on that cluster head.

Another consideration is that the algorithm has to give encouragement for clusters to grow, that is that the amount of nutrient available becomes greater as nodes join the network. To effect this in each phase the sink has an available amount of nutrient proportional to the total number of connected nodes. The starting conditions are as follows:

1. Some initial store of nutrient available at the sink
2. Current state of all other nodes is to have no nutrient

From the initial state, some nodes will by chance have direct contact with the sink. These become the initial cluster heads, and each is given an equal share of the nutrient available, which is available to them as attenuated by the square of the distance from the sink. Across the network the sequence of events in each phase is as follows:

Nutrient allocation: Each node transmits to its dependents (if any) the total amount of nutrient available to that cluster and the current number of members (including the cluster head) at that level of the hierarchy. Each dependent calculates its share of nutrient for this phase, which is

$$\frac{n}{m \times k \times d^2}$$

where n is the total nutrient available to the cluster, m is the number of members, k is a constant of proportionality for the distance adjustment and d is the distance between the node and the cluster head.

Cluster head advertisement: Each node which has a supply of nutrient selects another node (or set of nodes, to speed up the evolution of the system) at random and forwards the above information, along with the identity of the cluster head.

Nutrient estimation: The receiving node calculates the amount of nutrient it could have received in this phase as a member of that cluster. If the amount is greater than its actual allocation in this phase it communicates with the cluster head and joins the cluster (also communicates with its old cluster head to leave that cluster).

Cluster head propagation: Cluster heads propagate upwards through the network the number of members. The sink calculates the amount of nutrient available for the next phase using the formula

$$nn = \frac{no \times mn}{mo}$$

where nn is the nutrient available for the next phase, no is the nutrient available this phase, mn is the membership reported, mo is the membership reported for the previous phase.

6.5 Evaluation Metrics

In this work, the performance of MuMHR routing algorithm is to be evaluated via simulations with respect to the following metrics:

Data Delivery Ratio

Data Delivery Ratio (DDR) is a service level parameter that indicates the network effectiveness in transmitting offered data in one direction of virtual connection (Dunn and Martin, 2001). It is considered as one of the prime measures of reliability. DDR is a ratio of successful distinct payload octets received to attempted payload octets transmitted (Dunn and Martin, 2001). DDR was used in (Dubois-Ferriere and Estrin, 2004; Vidhyapriya and Yu, 2007; Sun et al., 2007) among others to measure the operating quality of sensor networks routing techniques. This measure is easy to obtain and free with every received packet.

DDR is an important metric because it reflects the congestion level of the network. It measures the protocol performance from loss ratio experienced at the network layer that is affected by factors such as packet size, network load, and topological changes. Higher DDR means that the packet loss rate is lower and the protocol is more efficient from the perspective of data delivery. When calculating DDR, the packets which arrived late at the destination are considered ineffective.

This ratio can be used to describe both individual sensors and the entire sensor network. The DDR for a single sensor S_i is denoted as $DDR(S_i)$ and defined as:

$$DDR(S_i) = \frac{\sum \text{data delivered to the sink}}{\sum \text{data offered by } S_i} \times 100\% \quad (6.1)$$

where *data delivered* is successfully delivered payload octets and *data offered* is the attempted payload octets transmitted. Distinctiveness of received packets is important because one packet may have different copies arriving at the sink. The overall DDR of a sensor network with a number of nodes n , denoted as DDR_N , is the average DDR of all sensor nodes:

$$DDR_N = \frac{1}{n} \sum_{i=1}^n DDR(S_i) \quad (6.2)$$

Timeliness

Timeliness is measured as the time normalised against the average time for a single-hop along the shortest path from a sensor to the sink (Sun et al., 2007). Recent studies in sensor networks focus on timeliness as a QoS metric (e.g., RAP (Lu et al., 2002), Implicit EDF (Caccamo et al., 2002), and SPEED (He et al., 2003)). Timeliness is an important metric because it measures the effect of aggregation on the routing protocol performance. While data aggregation helps in reducing the number of packet transmissions, it increases the queueing delay at the cluster heads due to increased inbound traffic and waiting time for the arrival of the data packets to be aggregated. The increased queueing time can risk the timeliness of constrained traffic. Timeliness only considers those packets that are successfully delivered from the source to the sink. The average delay taken by the first copy of a packet from the source node, S_i , to the sink is denoted as $T(S_i)$. This delay includes all possible delays that are caused by queueing in the interface queue, retransmission at the MAC layer and the propagation through the environment. The average delay of all n sensors, denoted as T_N , is given by:

$$T_N = \frac{1}{n} \sum_{i=1}^n T(S_i) \quad (6.3)$$

where

$$T(S_i) = (\text{time a packet arrives at the sink}) - (\text{time a packet is sent at the source})$$

The delay does not only depend on the routing mechanisms, but also on the scale of the network (Sun et al., 2007). The average network delay is the total delay divided by the number of connection pairs throughout the simulation to eliminate the effect of the network scale. Then timeliness is measured as follows (Sun et al., 2007):

$$T_{Network} = \frac{T_N}{\frac{1}{n} \sum_{i=1}^n h(S_i)} = \frac{\sum_{i=1}^n T(S_i)}{\sum_{i=1}^n h(S_i)} = \frac{1}{n} \sum_{i=1}^n \frac{T(S_i)}{h(S_i)} \quad (6.4)$$

where $h(S_i)$ is the minimum hop count from S_i to the sink. $T_{Network}$ is the average time it takes a packet to be delivered from a source node to the sink. When the value of $T_{Network}$ is 1, it means that every packet is delivered on the minimum number of hops path.

Energy Efficiency

The main purpose of MuMHR is to extend network life time. Network lifetime can be defined as the time elapsed until either the first node dies, half the nodes die, or the last node dies (Younis and Fahmy, 2004). A node is defined as dead after its remaining energy drops to a pre-set threshold. Besides the network lifetime, the total network energy available after each cluster formation round is defined as a sub-metric to reveal the total energy consumption in transmission and reception of all packets in the network. This metric alone shows how efficient the algorithm is with respect to energy consumption.

MuMHR also aims at reducing the setup communication overhead to achieve more energy savings as communication is the most power hungry operation. Pottie and Kaiser (2000) calculated the energy consumption and found that a general-purpose processor could execute 3 million instructions for the same amount of energy used to transmit 1 Kbit of data by radio over 100m. Thus, the cost that MuMHR imposes on the networks is considered in terms of number of messages sent. This measure is important because it gives an indicator about the bandwidth usage besides the energy consumption. In this study we are only interested in studying the effect of using the back-off time and the number-of-hops metric on the cluster setup traffic. Therefore, the setup traffic for a single cluster formation round, Tr_{Round} is calculated as:

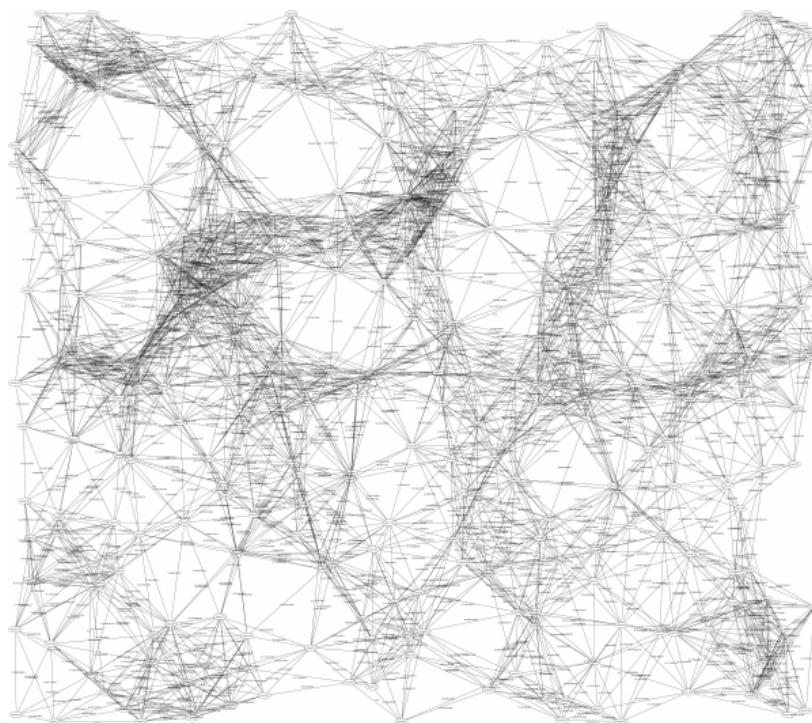


Figure 6.7: A 100 nodes wireless sensor network with random topology

$$Tr_{Round} = \frac{\text{number of sent messages}}{C_{life}} \quad (6.5)$$

where C_{life} is the time spanning from the moment the cluster head sends the first advertisement message to the moment all cluster members receive the TDMA schedule.

6.6 Simulation Experiments

As MuMHR inherits its basics from LEACH, we adopt the same network and energy model for better comparison. Random graphs were dispersed in a $100m \times 100m$ region such that no two nodes share the same location and the transmission range of each node is bound to $75m$. The bandwidth of

the channel was set to $1Mbps$, each data message was 500 bytes long, and the packet header for various message types was fixed to 30 bytes. A simple model for radio hardware energy dissipation is also assumed. The transmitter dissipates energy to run the radio electronics and power amplifier and the receiver dissipates energy to run the radio electronics. All the nodes were given an initial 0.5J supply of energy. For the experiments described here, both the free space (d^2 power loss) and the multi-path fading (d^4 power loss) channel models were used. The processing delay for transmitting a message is randomly chosen between 0 and 5ms.

This experimental work is based on the hardware platform provided by the Crossbow Technology Inc. The Mica2/Dot professional mote kit (2007). Figure 6.7 shows a random node distribution topology of 100 nodes. Using this network configuration, we simulated LEACH and MuMHR with 5% of the nodes being cluster heads.

MuMHR Evaluation in Dingo

MuMHR and LEACH were implemented in the Dingo wireless sensor network simulator (Mount, 2008). Additionally, MuMHR was implemented for the NS2 simulator and the LEACH author's implementation of their protocol was used as a comparison reference. We use both simulators since each simulator highlights different aspects of each protocol. For example, Dingo does not implement packet collisions while NS2 puts restrictions on the sensing hardware model which limits its flexibility. However, in our experiments we only use Dingo simulation results for LEACH since its NS implementation released by its authors has several bugs especially in the energy measurements.

The difference in results generated by Dingo and NS2 is due to the variations in the simulated network topologies as well as the collision management strategies in both simulators (see Section 2.4). Also the queueing delays in Dingo cause some packets to arrive late and thus they are counted

as lost packets. Particularly, Dingo uses a Scheduler API to schedule execution of a list of send and receive tasks. The Scheduler API only runs one task at a time leaving no chance for collisions to happen. On the other hand, NS2 uses TDMA to schedule messages transmissions.

Experiment 1: Cluster Size

Aim: This experiment aims to study the effect of introducing the number-of-hops and the back-off waiting time metrics on the cluster formation process.

Procedure: Four network topologies with equal number of nodes were generated by different simulation runs. In each topology, nodes are organised into five clusters with $P = 0.05$ where P is the desired percentage of cluster heads. Then, node distribution among different clusters is observed. The lines indicate the borders of different clusters. Topologies (a) and (b) were generated using LEACH, whereas topologies (c) and (d) were generated using MuMHR with the back-off waiting time set to 20 seconds and the number-of-hops initially set to 0.

Results and discussion: Figure 6.9 shows the node distribution among clusters in the four random network topologies. The graphs compare the distribution of nodes among clusters formed using LEACH with those formed using MuMHR. In the first LEACH topology (a), the first and fifth clusters hosted over 65% of the total number of nodes while the other three clusters hosted less than 35% of the total network population. In the second LEACH topology (b), the fifth cluster had 0 nodes while the percentage of nodes fluctuated among the other clusters between 7% and 30%. It can be clearly seen that there is uneven distribution of nodes amongst the clusters, which increases both the cluster management overhead and the energy consumption. In both topologies (a) and (b), the area covered by

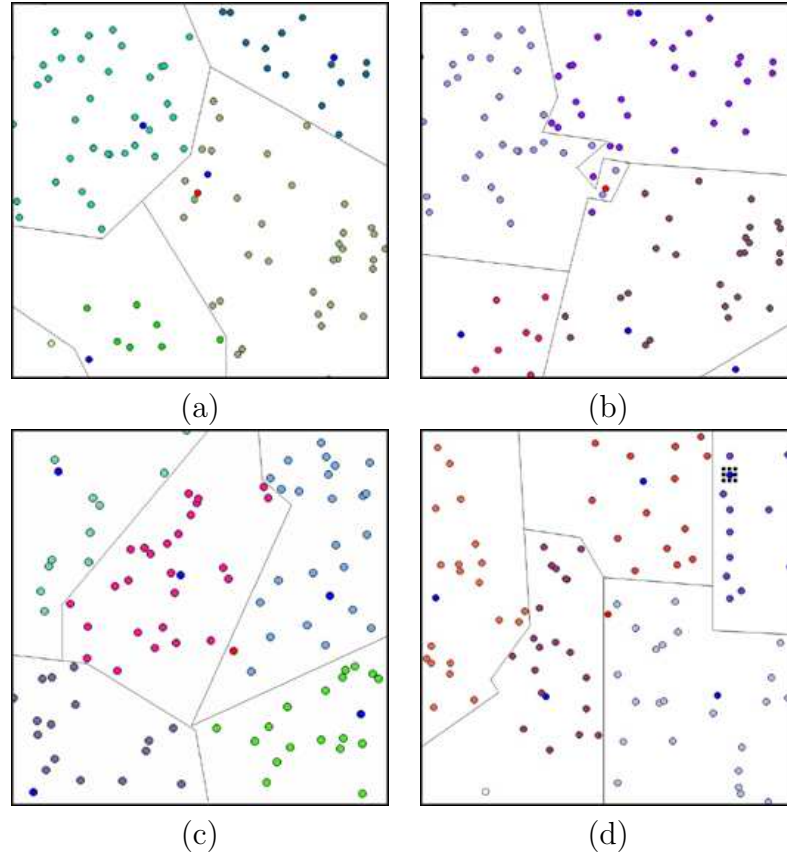


Figure 6.8: Geographically uniform cluster formation.

different clusters varies largely. These cluster topologies are not energy efficient because data needs to traverse a long distance to reach the cluster head. Besides, the load is not evenly distributed among different cluster heads in the network. Whereas in MuMHR topologies (c) and (d), nodes were distributed much more fairly among clusters. The number of nodes at every cluster maintained a maximum of 7% difference from the optimal population (20%). In MuMHR generated topologies (c) and (d), the area covered by each cluster is almost equivalent and nodes are distributed more fairly among the five clusters than in topologies (a) and (b) as seen in Figure 6.8. The efficient distribution of a sensor network into uniform, mostly

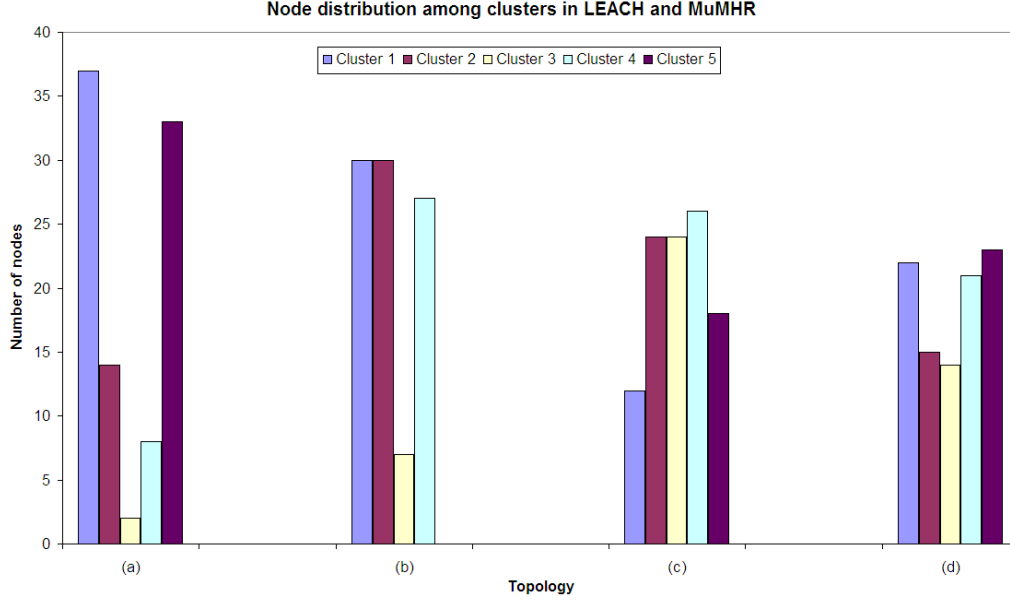


Figure 6.9: Node distribution among clusters formed by MuMHR.

non-overlapping clusters of physically close nodes is a central element in the design of efficient higher level network functions such as the mapping service.

Conclusion: The results given in figures 6.9 and 6.8 demonstrate that the back-off waiting time together with the number-of-hops metric lead to the formation of more energy efficient clusters by shortening the communication routes. Given the number of clusters in all network topologies studied in this experiment is 5, then the optimal coverage area is 20% of the monitored terrain area. The average variance, from the optimal, of the coverage polygon of the 10 clusters in MuMHR network topologies (c) and (d) is 3.1075. This variance is approximately 3 times smaller than that of the 10 clusters generated by LEACH (11.818). The back-off waiting time gives more time to receive a smaller number-of-hops value. This allows nodes to register with the closest cluster head resulting in more geographically uniform clusters. In MuMHR the number of nodes in each cluster is within a narrow

range (7%).

Experiment 2: Fault-Tolerance and Reliability

Aim: This experiment aims to measure MuMHR's ability to maintain high delivery ratio of data packets within a given timeliness constraint in spite of some communication failures.

Procedure: MuMHR was simulated in Dingo and NS2 using random network topologies. Experiments were repeated with different network topologies, varying the number of nodes. Also, LEACH was simulated in Dingo and compared with the MuMHR algorithm.

Results and discussion: Figure 6.10 shows the delivery ratio of data packets for MuMHR and LEACH protocols measured using Dingo and NS2 simulators. The delivery ratios of the two routing protocols increase as the node density increases. In MuMHR, when node density is high there are more nodes available for data forwarding, and this increases the delivery ratio. Dingo and NS2 generated nearly similar delivery ratios for MuMHR which is very close to the ideal one. LEACH, however, has slightly more fluctuations. Figure 6.10 shows that LEACH offers less packet delivery rates, first is MuMHR; it did not adapt its behaviour well to the increase in network size. In MuMHR, the use of multi-path, multi-hop routing maintained constant delivery rates throughout the simulated scenarios. This is a result of the process it uses to create a routing path. Figure 6.10 also shows that MuMHR always achieves very high packet delivery ratios, close to 1 in all tested node densities. In MuMHR, sensing nodes send packets to their cluster head and the cluster head forwards packets to the sink; thus there is little congestion. In LEACH, cluster formation occurs frequently and results in a larger communication overhead. Due to this energy depletion, sensors die quickly and cause packet loss. When the node density increases, the effect of some node failures on packet delivery ratio is reduced. This is why

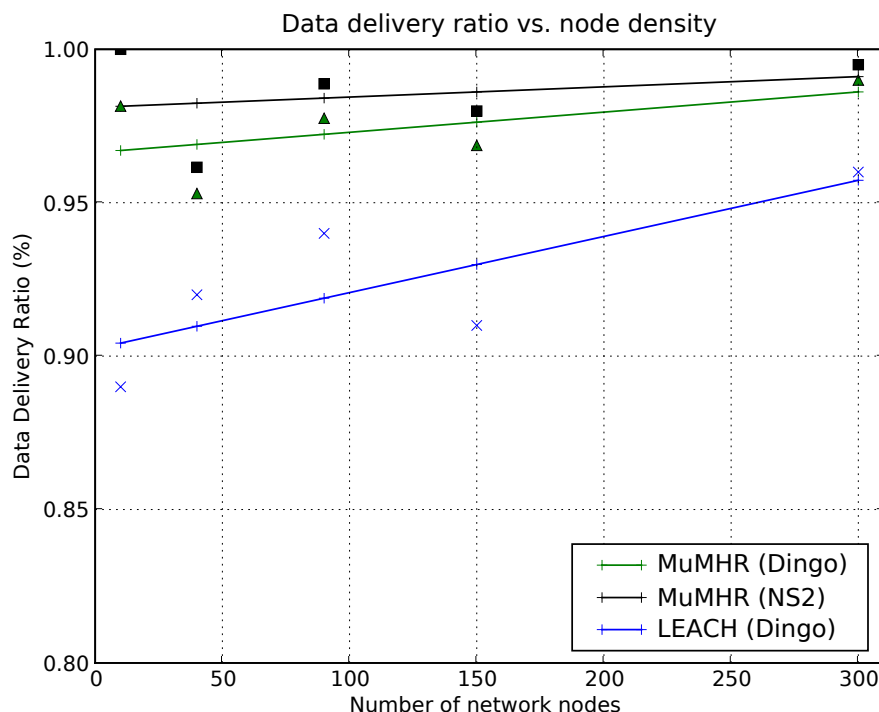


Figure 6.10: Comparison of data delivery ratio of MuMHR and LEACH as a function of node density in the network.

the packet delivery ratio in MuMHR increases as node density increases. In LEACH, a large number of sensors need to communicate directly with the sink and this may cause interference and congestion in the network, and thus causes more packets lost.

Figure 6.11 plots the average end-to-end delay for the MuMHR and LEACH routing algorithms at different node densities using the Dingo and NS2 simulators. At each plotted point, the end-to-end delays were averaged for at least six runs. Under all network node densities, Figure 6.11 shows that MuMHR routing algorithm has a short or average end-to-end delay in comparison to LEACH. There are several factors accounting for this outcome. First, the lossless aggregation used in LEACH can dramatically in-

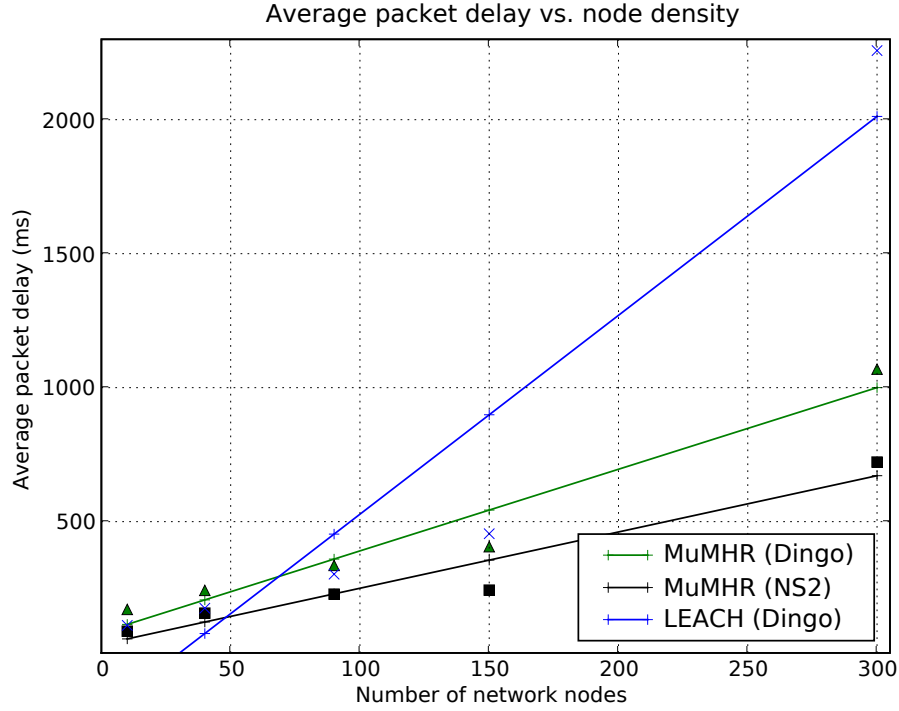


Figure 6.11: Average end-to-end packet delivery delay as a function of node density in the network.

crease the average packet end-to-end delay when the traffic becomes heavy. The cluster head in LEACH forwards its cluster data only after receiving the data packet from the last node in its broadcast TDMA schedule causing severe delivery delays and even loss for the whole cluster data. While the time aggregation function in MuMHR is always successful in keeping the delay below the bound, lossless aggregation is successful only to a certain extent, after which the cluster head becomes overloaded. MuMHR tends to forward data packets using the shortest path to the destination, thereby reducing the number of hops and the end-to-end delay. Furthermore, in time critical applications MuMHR multiplexes data over its acquired multi-paths to avoid queueing delays or congested paths.

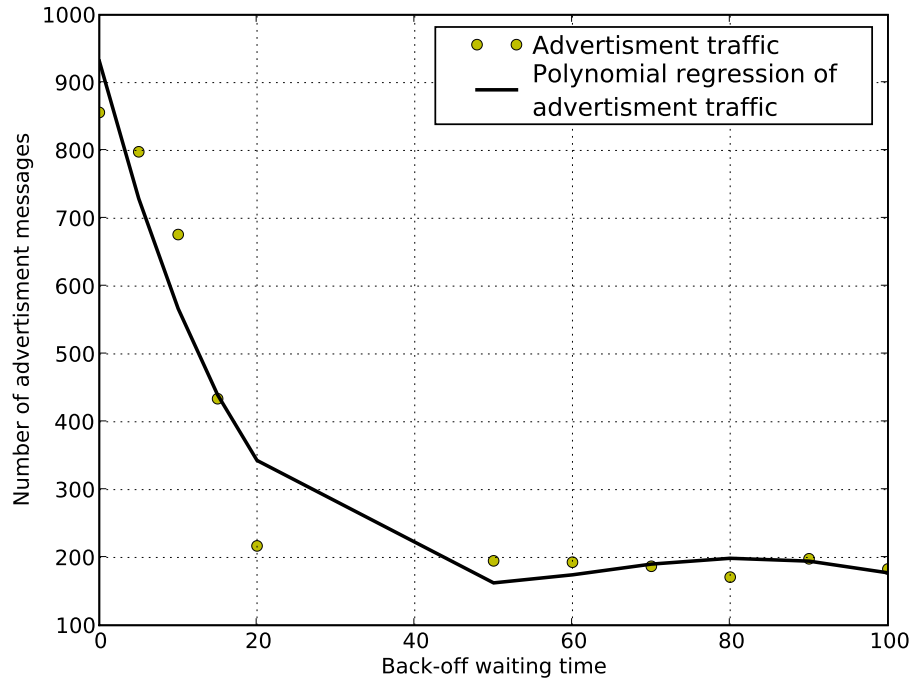


Figure 6.12: The number of sent messages versus the back-off waiting time in *ms*.

Conclusion: Unlike LEACH, MuMHR adapts its behaviour to increases in network size. Under all network densities, the use of multi-hop, multi-path routing helped MuMHR to maintain constant data delivery rates and shortened the average end-to-end delay in comparison to LEACH.

Experiment 3: Energy Consumption

Aim: This experiment aims to study the effect of the back-off waiting time and the number of hops parameters on the energy consumption.

Procedure: Because Dingo has inefficient energy module and the NS2

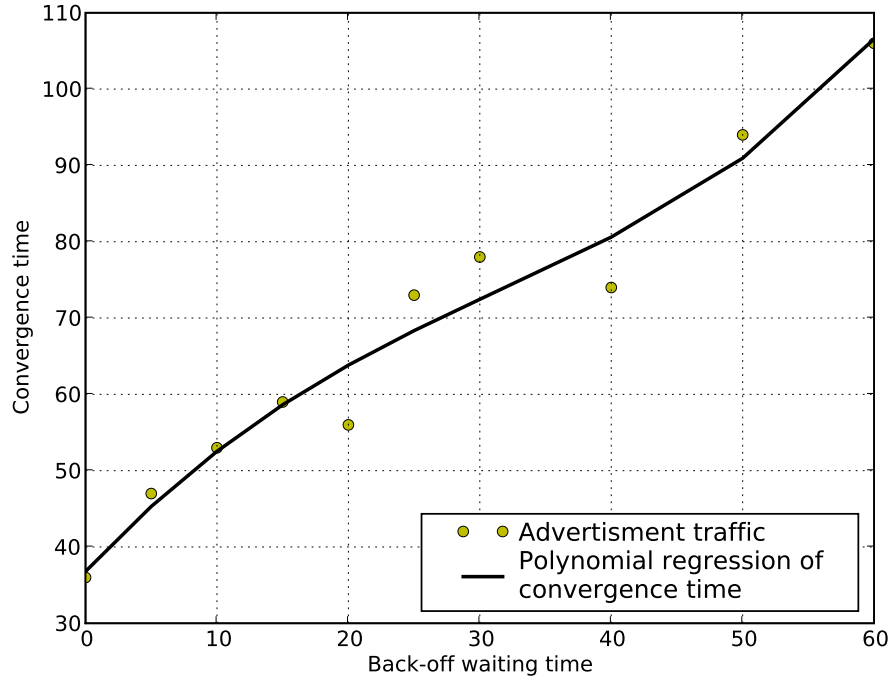


Figure 6.13: The convergence time versus the back-off waiting time in *ms*

implementation of LEACH was found to produce incorrect results, this work only includes NS2 energy simulation results for MuMHR. MuMHR simulations were repeated varying the back-off waiting time.

Results and discussion: In Figure 6.12, the number of network setup messages versus the back-off waiting time is plotted. When the back-off waiting time is set to 0 the total number of sent messages will be similar to that in LEACH. The figure shows that as the back-off waiting time becomes larger, the number of messages will decrease until the time becomes large enough to receive advertisements from all cluster heads. The optimal back-off waiting time is calculated from Figure 6.12 to be around *20ms*. Using these two parameters, the total number of set-up messages is reduced by

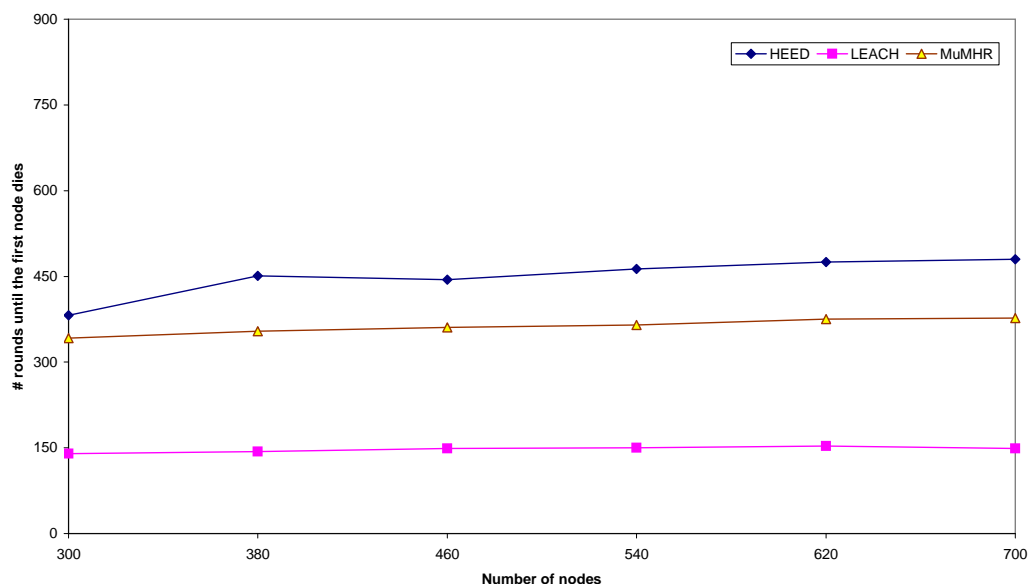


Figure 6.14: MuMHR network lifetime (first node death) compared to HEED and multi-hop LEACH lifetime results published in (Younis and Fahmy, 2004).

up to 65%.

Figure 6.13 plots the network convergence time versus the back-off waiting time. A linear correspondence between the time needed to establish routes and the back-off waiting time is evident. As the back-off waiting time increases, network convergence time increases proportionally. In time critical applications it is important that the convergence time is reduced. This means, reducing the back-off waiting time parameter to a minimal value to capture the advantages of this parameter with minimal delay to achieve efficiency.

When the number of rounds until the last node dies is measured with the back-off time set to 0 and the number of hops is always set to 1, the behaviour of MuMHR gets very close to LEACH. Figures 6.14 and 6.15 show MuMHR network life. In Figure 6.14 the network life is defined as the time until the first node dies while in Figure 6.15 it is defined as the time until

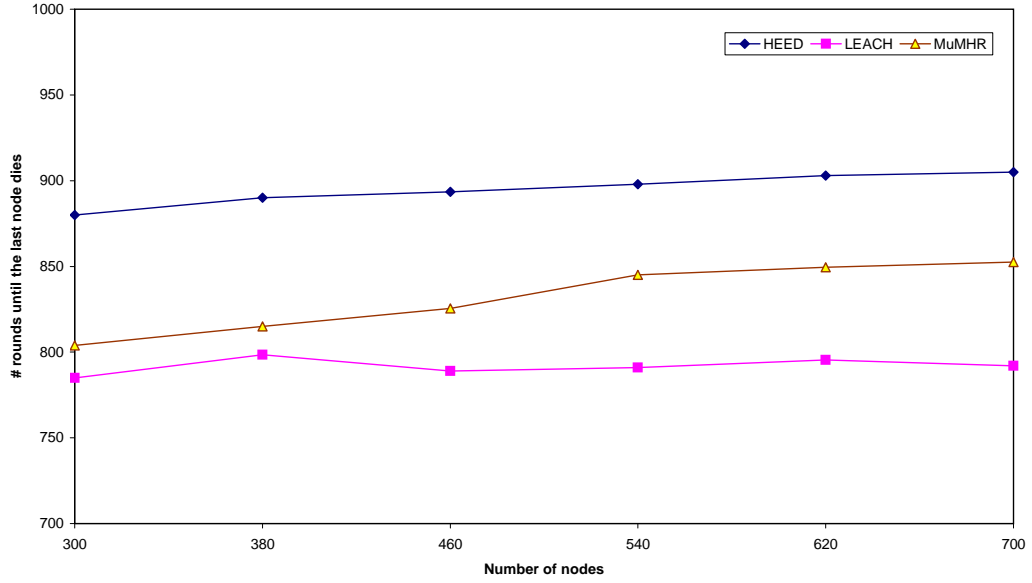


Figure 6.15: MuMHR network lifetime (last node death) compared to HEED and multi-hop LEACH lifetime results published in (Younis and Fahmy, 2004).

the last node dies. Looking at these figures, MuMHR parameters, back-off time and the number of hops, clearly improves network life. This is because the communication cost is minimised by reducing clustering overhead and by finding the shortest multi-hop path from the source to the destination. Finally, Figures 6.14 and 6.15 show that MuMHR can achieve comparable energy levels but more balanced performance when compared to results published in (Younis and Fahmy, 2004).

Conclusion: This experiment proves that the back-off waiting time is effective in reducing the overall set-up energy consumption by only forwarding the best received cluster advertisement message. The back-off waiting time must be reduced to a minimal value to capture the advantages of using this parameter with minimal cluster setup delays. MuMHR can achieve comparable energy levels but more balanced performance when compared to LEACH.

6.7 Chapter Summary

In this chapter, an improvement over LEACH called MuMHR was presented. MuMHR provides solutions to some of the limitations of LEACH. It can achieve comparable energy complexity but more balanced performance by deploying the number-of-hops and the back-off waiting time parameters. The algorithm uses redundant messages received from different sources to build a multi-path map, which allows auto-adaptation to path failures. MuMHR also enables multi-hop transmissions to relax LEACH's inflexible assumption that all nodes in the network can communicate with each other. The new algorithm achieves robustness and efficiency without location information and with comparable energy expenditure to LEACH. However, MuMHR is open to utilise the location information if available. Simulation results confirm that the protocol incurs low overhead in terms of processing cycles and messages exchanged. It also achieves fairly uniform node distribution across the clusters in the network. Simulation results also show that MuMHR extends network lifetime, and the clusters it forms show several appealing features. MuMHR parameters, such as the number of hops and the back-off waiting time, can be easily adjusted to optimise resource usage according to the network density and application requirements. This routing algorithm was implemented and used for the mapping service and found to easily support various computationally demanding mapping applications for wireless sensor networks.

Chapter 7

A New Network Service: Map Generation

Not all information that is collected from a wireless sensor network comes ready to use. Often, wireless sensor networks field data collection takes the form of single points that need to be processed to get a continuous data presentation. Interpolation describes this process of taking many single points and building a complete surface, the inter-node gaps being filled based on the spatial statistics of the observation points. Interpolating these points will produce more useful information for the end user such as chemistry maps, and maps related to water chemical content, for example the degree of nitrate contamination. The ability to interpolate point information is necessary for carrying out mapping tasks.

The problem of map generation is essentially a problem of interpolation from sparse and irregular points. This interpolation capability is realised as a service of the network. In this chapter, one particular interpolation approach, Shepard interpolation (Shepard, 1968), is examined and shown to be suitable for the constraints imposed by the nature of wireless sensor networks. Visual aspects, sensitivity to parameters, and timing requirements were used to test the characteristics of this method

7.1 Introduction

Visualisation can be seen as a process of visual reconstruction (Brodliet al., 2005). Most scientific visualisation techniques involve an interpolation step (Pang et al., 1994). Interpolation is the process of estimating the value of sense modality at un-sampled sites within the area covered by existing observations. It defines a function that takes on specified values at specified points. There are different ways for defining that function, many of which involve fitting some sort of function to data and evaluating that function at a desired point. There are also other means of calculating interpolated data, such as statistical methods.

In this chapter, one of the first distance weighting interpolation algorithms for irregularly spaced data, known as the Shepard interpolation algorithm (Shepard, 1968), is studied in detail and compared to Triangulation with Linear Interpolation (TLI) (Yang et al., 2004). Shepard and TLI are important interpolation algorithms that are used in many practical wireless sensor networks and virtual reality projects (Kreylos, 2007).

7.2 Shepard Interpolation Algorithm Details

Inverse distance weighted interpolation algorithms are probably the oldest known scattered data interpolation algorithms (Lee et al., 1997; Ruprecht and Müller, 1994). One such algorithm was originally proposed by Shepard and has become known as Shepard's algorithm. Shepard interpolation is widely used in practise and has also been shown to work well with noisy data (Jr. and Bolstad, 1996). Shepard defined a continuous function where the weighted average of data is inversely proportional to the distance from the interpolated location. The algorithm explicitly implies that the further away a point is from an interpolated location, P , the less effect it will have

on the interpolated value. This algorithm exploits the intuitive sense that things that are close to each other are more alike than those that are further apart.

The points, D_i , are weighted during interpolation relative to their distance from P . Weighting is assigned to data through the use of a weighting power which controls how the weighting factors drop off as the distance from P increases. The choice of weighting power parameter can significantly affect the interpolation results. The greater the weighting power, the less effect far points have on the interpolation result. As the power increases, Shepard interpolation approaches the nearest neighbour interpolation method (Yang et al., 2004) where the interpolated value simply takes on the value of the closest sample point (Yang et al., 2004; Jr. and Bolstad, 1996). For smaller weighting values, the weights are more evenly distributed among the neighbouring data points resulting in a smoother surface. When calculating the value of an intermediate point which is coincident with a known point, the intermediate point is assigned the value of the coincident point.

Inverse distance algorithms can be either exact or smoothing. With weighted average interpolators, exact interpolation means that the surface passes through all points whose values are known. Whereas, the smoothing parameter is used when there is some uncertainty about the given surface values. This utilises the fact that in many data sets there are global trends, which vary slowly, overlaid by local fluctuations, which vary rapidly and produce errors in the recorded values (Goodchild and Kemp, 1990). The purpose of the smoothing parameter will therefore be to reduce the effects of errors on the resulting surface by eliminating surface noise (Yang et al., 2004). Smooth surfaces are rendered by rounding angles which are less than the smoothing parameter. If unexplained rendering errors appear on the smoothed surfaces, then increasing the smoothing parameter could correct for such errors.

Many modifications to the original Shepard interpolation algorithm have been proposed in the literature (Iyer et al., 2006; Renka and Brown, 1999a,b; Berry and Minser, 1999), however, most of these methods are designed for computer graphics and image processing fields. These algorithms usually trade accuracy with computation complexity. Nevertheless, when applying interpolation to wireless sensor network applications it is desirable to keep the interpolation algorithm simple to reduce the amount of processing as well as communicated information across the network. Therefore, we shall use the original method with the modifications proposed by Shepard that further reduce the amount of processing and information communication to achieve more energy savings using the limited available bandwidth. These modifications are described in subsection 7.2.

Global Shepard Algorithm Description

Shepard's expression for globally modelling a surface is:

$$f_1(P) = \begin{cases} \frac{\sum_{i=1}^N (d_i)^{-u} z_i}{\sum_{i=1}^N (d_i)^{-u}} & \text{if } d_i \neq 0 \text{ for all } D_i (u > 0) \\ z_i & \text{if } d_i = 0 \text{ for some } D_i \end{cases} \quad (7.1)$$

where d_i is the standard distance metric from an interpolation point, P , to the point D_i numbered i in the N known points set and z_i is the known value at point D_i . The exponent u is used to control the smoothness of the interpolation. As the distance between interpolation location P and the measured sample point D_i increases, the weight of that sampled point will decrease exponentially. As P approaches a data point D_i , d_i tends to zero and the i^{th} terms in both the numerator and denominator exceeds all bounds while other terms remain bounded. Therefore, the $\lim_{P \rightarrow D_i} f_1(P) = z_i$ is

as desired and the function $f_1(P)$ is continuously differentiable even at the junctions of local functions.

The partial derivation of f_1 exists at all points. For $u > 1$, the continuity of the first derivative is guaranteed and its value approaches 0 at all D_i . When $u < 1$ no derivative exists, thus the requirement that the interpolation function be differentiable imposes that the exponent exceeds 1.

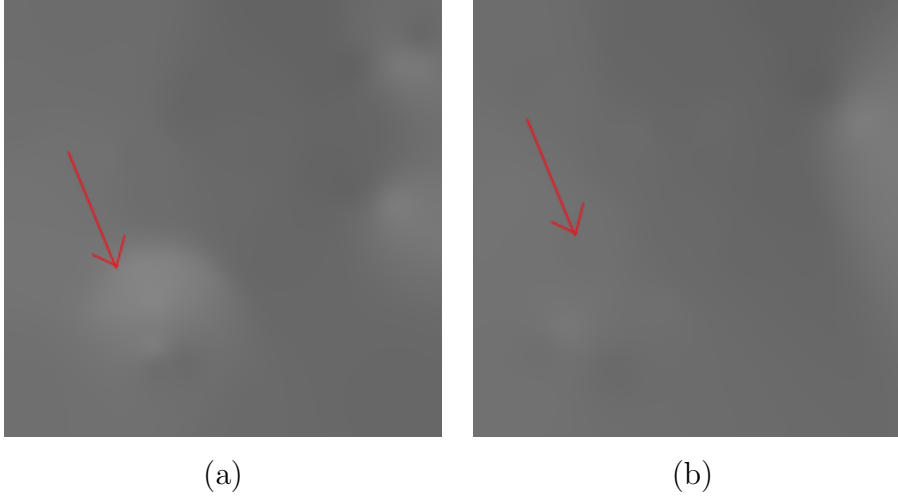


Figure 7.1: Figure (a) shows an example of Shepard interpolation bull's-eye effect. In Figure (b) this undesired effect was reduced by performing a smoothing step.

The optimal P value is determined by minimising the Root Mean Square Prediction Error (RMSPE) (Krivoruchko, 2004). The RMSPE is the statistic that is calculated from cross-validation. In cross-validation, each measured point is compared to the predicted value for that location (Krivoruchko, 2004). Experimental tests performed in (Shepard, 1968; Krivoruchko, 2004) revealed that exponent values greater than 2 tend to make the surface flat near all sampling points with high gradients over small intervals between sampling points. Lower exponents produce a relatively flat surface with short blips to attain the accurate values at sampling points. An exponent

of $u = 2$, when used for general mapping and description purposes, gives satisfactory experimental results and reduces computation cost. According to (Yang et al., 2004), when a non-zero exponent is assigned, no point is given an overwhelming weight, meaning that no point is given a weighting value equal to 1.0.

One of the characteristics of this method is the generation of “bull’s-eyes” surrounding the position of observation within the interpolation area. A smoothing exponent can be assigned during the interpolation process to reduce the “bull’s-eye” effect. Figure 7.1 (a) presents an example of bull’s-eyes effect generated by Shepard interpolation. This undesired effect was reduced by applying a smoothing process on the interpolated surface and the results are presented in Figure 7.1 (b). The smoothing parameter allows incorporating an uncertainty factor associated with the input data. The larger the smoothing parameter, the less overwhelming the influence any particular observation has in predicting a neighbouring grid node.

Global Shepard Algorithm Shortcomings and Solutions

Shepard’s interpolation algorithm is widely used due to its simplicity, but it suffers from several shortcomings imposed by the fact that each sample point has a radially symmetric influence despite the nature of the underlying data (Park et al., 2005). Among the well known artifacts are cusps, corners, and flat spots at the data points, as well as the excessive influence of points that are far away (Lee et al., 1997). Further shortcomings include that the global function necessitates all weights to be recomputed if any points are added, removed, or modified. In sensor networks this is impractical due to the network dynamics such as: node failures, node mobility, or deployment of new nodes. Shepard has identified four main shortcomings of the method, but this work deals only with the first three shortcomings as the forth shortcoming is not valid any more due to today’s high computer precision.

Shepard proposed modifications to the original method 7.1 as follows:

1. Building the Support Set: We define the *support set* as the set containing all points used to calculate P . The global method has a linear running-time $O(N)$, which makes it impractical and inefficient especially when the number of data points is large. The time complexity to calculate $z = f_1(P)$ grows linearly with the size of the input data set; if the input is twice as large, the algorithm takes roughly twice as long to calculate $z = f_1(P)$. To overcome this, a local Shepard algorithm was defined. This algorithm eliminates distant points from the calculation of any interpolated value since only nearby data points have significant influence. The interpolation from nearby nodes results in considerable computation savings and acceptable surface reproduction for many applications. To select nearby nodes, Shepard defined two criteria:

- a) Arbitrary distance criterion: All data points within radius r of the point P are included in computation. This is computationally easy but allows the possibility that there are no data points or a sufficiently large number of data points within the radius r . A collection of points, C_p , within a search radius r is defined as $C_p = \{D_i | d_i \leq r\}$ and $n(C_p)$ is the total number of data points in C_p .
- b) Arbitrary number criterion: Only the closest n data points are considered in the computation of any interpolated value. This approach ignores the relative location and spacing of the points and requires deep searching and complex ranking procedure for data points. In addition, it assumes that a single number, n , of interpolating points was optimal. If N is the total number of data points, then, a new collection of data points, C_p^n , is defined

as $C_P^n = \{D_{i_1}, D_{i_2}, \dots, D_{i_n}\}$ where $(n \leq N)$ and the subscripts i_j are defined such that $0 \leq d_{i_1} \leq d_{i_2} \leq \dots \leq d_{i_N}$.

Shepard has chosen a mix of the two criteria which combined their advantages. An initial radius r is defined depending on the overall density of data points such that seven data points are included on average in a circle of radius r . r is written as follows:

$$\pi r^2 = \frac{7A}{N} \quad (7.2)$$

where A is the area of the largest polygon enclosed by the data points.

A function $s_i = s(d_i)$ is defined to guarantee the local behaviour of the interpolating algorithm by calculating a surface model for any $d \leq r$, and which weights the points at $r \leq \frac{r}{3}$ more heavily:

$$s(d) = \begin{cases} 1/d & \text{if } 0 < d \leq \frac{r}{3} \\ \frac{27}{4r^2} \left(\frac{d}{r} - 1\right)^2 & \text{if } \frac{r}{3} < d \leq r \\ 0 & \text{if } r < d \end{cases} \quad (7.3)$$

where r' is a radius of influence about point P chosen large enough to include n points and defined as $r'(C_P^n) = \min\{d_{i_j} | D_{i_j} \notin C_P^n\} = d_{i_{n+1}}$. In order that the interpolation algorithm works realistically, if the data points were girded, a minimum of four data points was chosen. A maximum of ten was established to limit the complexity and amount of computation required (Shepard, 1968). Thus C'_p and r'_p are defined as follow:

$$C'_p = \begin{cases} C_p^4 & \text{if } 0 \leq n(C_p) \leq 4 \\ C_p & \text{if } 4 < n(C_p) \leq 10 \\ C_p^{10} & \text{if } 10 < n(C_p) \end{cases}$$

and

$$r'_p = \begin{cases} r'(C_p^4) & \text{if } n(C_p) \leq 4 \\ r & \text{if } 4 < n(C_p) \leq 10 \\ r'(C_p^{10}) & \text{if } 10 < n(C_p) \end{cases}$$

The resulting function $f_2(P)$, has similar behaviour to the original function but it is capable of handling much larger data sets and it is much more suitable for parallel implementations. The interpolation function $f_2(P)$ is given by:

$$f_2(P) = \begin{cases} \frac{\sum_{D_i \in C'} (s_i)^2 z_i}{\sum_{D_i \in C'} (s_i)^2} & \text{if } d_i \neq 0 \text{ for all } D_i \\ z_i & \text{if } d_i = 0 \text{ for some } D_i \end{cases} \quad (7.4)$$

The function $f_2(P)$ requires less physical resources, in terms of both computation and memory, than the previous functions. The running-time is reduced to $O(C'_p)$ by only considering the set of data points $n(C'_p)$ within some radius r' . Consequently, the amount of memory used by the algorithm on data set of size n is reduced to inputs of C'_p size. This reduction in computation and memory cost is invaluable in large scale wireless sensor networks which must be capable of in-network processing at all levels, including the application level. For example, in the mapping application discussed in subsection 5.3, the interpolation algorithm uses the data set as vertices in the interpolated map to calculate an intermediate point. The local algorithm restricts the data set only to points that have considerable effect on the calculated point. This results in significant savings in communication, memory, and computation.

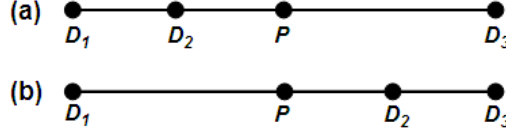


Figure 7.2: Two configurations of co-linear points adopted from (Shepard, 1968).

2. Including Direction: The current method only considers the distance factor and ignores the direction factor in computing the weightings. To make the method intuitively reasonable, Shepard included the direction in computing interpolated values. To clarify the problem consider the two configurations of co-linear points in Figure 7.2 adopted from (Shepard, 1968). The two configurations (a) and (b) will result in identical interpolated values at P if the direction factor is ignored. These results are not reasonable because an intervening data point should be expected to reduce the effect of the more distant point. However, the interpolated value at P in configuration (a) should be closer to the value at D_3 than in configuration (b) and conversely for the value at D_1 .

To improve the method, both the direction and distance from point P to the data points D_i were considered. A new directional weighting for each data point D_i close to P is defined by:

$$t_i = \sum_{D_j \in C'} s_j [1 - \cos(D_i P D_j)] / \sum_{D_j \in C'} s_j \quad (7.5)$$

were the $\cos(D_i P D_j)$ is defined as: $[(x - x_i)(x - x_j) + (y - y_i)(y - y_j)] / d_i d_j$. The appropriateness of the cosine function and computation ease makes it a good measure of direction. The function s_j is included in the new function to preserve the original implicit assumption that points D_i near P should weight more than distant points. Within the

direction considered, a new weighting function $w_i = (s_i)^2 \times (1 + t_i)$ is defined and the final interpolation function is defined as:

$$f_3(P) = \begin{cases} \sum_{D_i \in C'} w_i z_i / \sum_{D_i \in C'} w_i & \text{if } d_i \neq 0 \text{ for all } D_i \\ z_i & \text{if } d_i = 0 \text{ for some } D_i \end{cases} \quad (7.6)$$

The latest version of the function behaves like $f_2(P)$ with the direction factor included in computing the weightings. This directional weighting function accounts for intervening data points and avoids intuitively non-reasonable results.

3. Determining Slope: The arbitrary and undesirable zero gradient at every point D_i still exists on the $f_3(P)$, interpolated surface. If d_i is very small, s_i will equal d_i^{-1} and w_i will vary as d_i^{-2} . To correct this, weighted averages of divided differences of z_i about D_i , A_i and B_i , were added to sufficiently nearby data points to achieve partial derivatives at D_i . Constants A_i and B_i represent the slope in the x and y directions at each data point D_i , A_i and B_i are defined as:

$$A_i = \frac{\sum_{D_j \in C''_i} w_j \frac{(z_j - z_i)(x_j - x_i)}{(d[D_j, D_i])^2}}{\sum_{D_j \in C''_i} w_j} \quad (7.7)$$

and

$$B_i = \frac{\sum_{D_j \in C''_i} w_j \frac{(z_j - z_i)(y_j - y_i)}{(d[D_j, D_i])^2}}{\sum_{D_j \in C''_i} w_j} \quad (7.8)$$

where $C''_i = C'_{D_i} - \{D_i\}$.

A new parameter v is defined with the distance dimension to bound the maximum effect the slope terms may have on the final interpolated value. For a contour mapping application, v can be defined as:

$$v = \frac{0.1[\max\{z_i\} - \min\{z_i\}]}{\sqrt{\max\{(A_i^2 + B_i^2)\}}} \quad (7.9)$$

An increment $\Delta z_i = \frac{A_i(x-x_i)+B_i(y-y_i)}{\frac{v}{v+d_i}}$ is computed for each $D_i \in C'_P$ as a function of P to include the effect of the slope in interpolating values at P . Thus the latest version of the interpolation function is:

$$f_4(P) = \begin{cases} \frac{\sum_{D_i \in C'} w_i(z_i + \Delta z_i)}{\sum_{D_i \in C'} w_i} & \text{if } d_i \neq 0 \text{ for all } D_i \in C' \\ z_i & \text{if } d_i = 0 \text{ for some } D_i \in C' \end{cases} \quad (7.10)$$

In function $f_3(P)$, the interpolated surface has a zero gradient at every D_i . The function $f_4(P)$ behaves as $f_3(P)$ but corrects this undesirable property. Using $f_4(P)$, the interpolated surface would achieve the desired partial derivatives at D_i .

Interruption of the Relationship Between Adjacent Points

Shepard interpolation is based on the intuitive assumption that there is a logical relationship between adjacent points. This assumption is, however, violated if some barrier, such as a river, ruptures the continuity of the surface (Shepard, 1968). The effect of physical barriers can be simulated easily due to the distance-dependent interpolation by including virtual barriers. The user may specify discontinuities in the metric space in which d_i is calculated using a different selection set of nearby data points and different weightings and slopes are being calculated. Given a detour of length $b[P, Di]$

perpendicular to the line between P and D_i , Shepard interpolation defines the effective distance to travel between the two points as:

$$d'_i = \{(d[P, D_i])^2 + (b[P, D_i])^2\}^{\frac{1}{2}} \quad (7.11)$$

where $b[P, D_i]$ is the strength of the barrier. When a barrier exists, d'_i replace d_i in all calculations. Whereas, if there is no barrier between D_i and P , $d_i = d_i$ and $b[P, D_i] = 0$. The effective distance is discontinuous as P crosses a barrier which result discontinuous interpolated surface at an obstacle. The inclusion of barriers in the interpolation will result in the selection of a different set of nearby data points, weightings, and slopes.

7.3 Shepard Interpolation Analysis

In this section the effectiveness of the Shepard interpolation algorithm is verified and its characteristics are studied quantitatively and qualitatively. The Shepard interpolation algorithm was compared with Triangulation with Linear Interpolation algorithm (TLI) (Yang et al., 2004) in order to assure its effectiveness.

TLI was chosen as a comparison framework because it is an exact interpolator which uses the optimal Delaunay triangulation. Delaunay triangulation is used extensively in the field of wireless sensor network. Uses include: adaptable network deployment (Wu et al., 2006), network coverage (Naznin and Nygard, 2006), locating and bypassing routing holes (Fang et al., 2004), distributed area computation (Greenstein et al., 2004), position-aware routing (Raweb, 2004), and spatial clustering (Virrankoski, 2005). Furthermore, TLI is widely referred to in the literature including in the image processing field (Ruprecht and Muller, 1995) and reported to be one of the simplest and most efficient algorithms with a good running time (Yang et al., 2004; Goncalves, 2006).

Triangulation with Linear Interpolation Algorithm Details

The TLI algorithm (Yang et al., 2004) is an exact interpolator that uses the optimal Delaunay triangulation. This method connects data points to form triangles that do not intersect with each other. The result of this process is a patchwork of triangular faces over the extent of the grid. The slope and elevation of the triangle is determined by the original data points defining the triangle and all nodes within the triangular plane are defined by the triangular surface. Since the triangles are determined by the original data, the data must be sampled at a high rate. TLI is fast with all data sets but it is not effective with few points (Yang et al., 2004). One advantage of triangulation is that, with enough data, triangulation can preserve break lines defined in a data file. For example, if a fault is delimited by enough data points on both sides of the fault line, the surface generated by triangulation will show the discontinuity (Yang et al., 2004).

Comparison Metrics

To determine the accuracy of the interpolation quantitatively, the Root Mean Square Deviation (RMSD) of a high resolution source data and the result of the interpolation using a subset of that data has been chosen as a measure of how similar two images are. If the images are exactly identical, this value is zero.

The *skewness* and *kurtosis* were also chosen to give a measure of the shape. Skewness and kurtosis are the most common statistical measures for deviation. They allow quantising the shape deviation produced by different methods. The kurtosis is a measure of the peakedness of a real-valued random variable where a high kurtosis distribution has a sharper peak and fatter tails and low kurtosis distribution has a more rounded peak with wider shoulders (Joanes and Gill, 1998). Skewness is a measure of the asymmetry

of the probability distribution of a real-valued random variable (Joanes and Gill, 1998). A distribution could have two kinds of skewness; positive skew or negative skew, where the mass of the distribution is concentrated on the left of the figure or the right of the figure respectively.

Visually, we use 2D and 3D height maps to determine the global accuracy of the interpolation method. The 2D maps depict the height where the pixel intensities depict depth values. Further qualitative accuracy assessments are done using empirical peak profiling to obtain peak information for studying the two algorithms local behaviour.

Experimental Setup

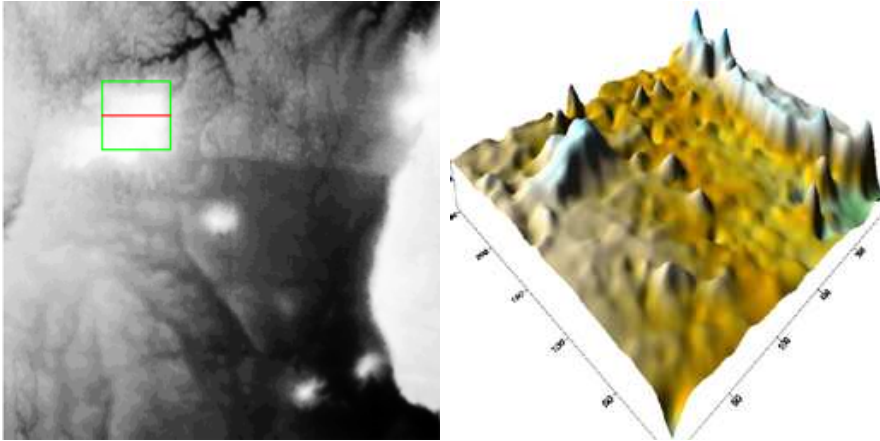


Figure 7.3: Grand Canyon height map

These experiments made use of the Grand Canyon height map (McCabe, 1998) to experimentally study the performance of TLI and Shepard interpolation algorithms. The studied region is $15360m^2$, with heights ranging from $165m$ to $284m$ above sea level. It is assumed that the deployment of data collection nodes is un-engineered, i.e. the data is irregularly distributed over the studied region. The height map was chosen because using

numerous wide-distributed height points has been an important topic in the field of spatial information (Yang et al., 2004). Furthermore, the height is a static measure which makes it suitable for the evaluation of various interpolation algorithms. However, the height is only one possibility for interpolation, nevertheless, interpolators may be used to estimate any sensed phenomena such as temperature.

The primary purpose of these experiments is to take spatial interpolation to calculate the unknown heights by using the information of neighbouring points and to report results and comparisons. Using the same data set, the difference in quality and accuracy of generated maps is determined by the interpolation method used.

The performance of interpolation algorithms is largely dependent on the characteristics of the input data set, such as accuracy, density, and distribution. The interpolation problem of randomly scattered data will be addressed because in many real-life deployments of wireless sensor networks, the position of sensor nodes is not predetermined. Ganesan et al. (2004) outlines two fundamental reasons that render nonuniform regular configurations of sensor network deployment.

1. The monitored phenomena are not uniformly distributed and the deployment of sensor nodes will be variable in order to achieve denser sensing where there is greater spatial variability.
2. Terrain and other deployment practicalities bias deployment locations to where necessary power sources, communication, or access can be achieved. Sensor networks are affected by the terrain condition. For example, the terrain could be inaccessible due to biological or chemical contamination, beyond the enemy lines, or even simpler, when the monitored area is very large.

Experiment 1: The Effect of Network Density

Aim: In this set of experiments, the effect of network density on the reconstruction quality of both interpolation algorithms is to be studied.

Procedure: The run of the interpolation methods with different network densities provides an excellent example of a framework from which to consider the network density problem.

Results and discussion: Tables 7.1 to 7.4 show how the network density and choice of interpolation algorithm affect the reconstruction results. It is observed that higher network densities increase the smoothness in the re-constructed maps. For instance, contour maps made from high network density are visibly smoother than contour maps made from lower density due to shorter line segments between data points. However, an increase in the number of data nodes results in an increase of the computation time, communication cost, and memory usage. Thus, the aim is to have an interpolation algorithm that is capable of generating a high quality map without having to obtain unrealistic percentages of data coverage in the area of interest.

Compared to the actual height map, Figure 7.3, it is visually evident that both interpolation algorithms produced acceptable quality 2D and 3D height maps. However, the reconstruction quality of TLI with the absence of sufficient data density (e.g. 10, 50, and 100 node) is largely fictitious and unconstrained especially on the map boundaries. TLI requires the number of boundaries of the observed area and higher density of sensing nodes on these locations. As the network density increases the reconstruction quality for both algorithms is improved. TLI performed better than Shepard with the reconstruction of the right hand side portion of the map due to the smoothness of its surface. This result is due to the assumption that TLI makes, that the height is changing at constant rate, which was the case in that portion of the map. Nevertheless, Shepard interpolation results were equal to or better than TLI algorithm despite the little geometric variation

in that part of the map. Shepard interpolation captured smaller features of the surface and reflected more details than TLI, as shown in Experiment 2. However, the cost (in terms of computation, the size of the support set, and the communication cost to collect the values in the support set) of interpolation in Shepard was much less than that when using TLI. Shepard reduces the interpolation cost by reducing the size of the support set to include only the points which have significant effect on P .

Conclusion: Shepard interpolation resulted in equal or better reconstruction results than TLI. The Shepard algorithm proved to produce more accurate interpolation results especially on the boundaries and at low network density. Also, Shepard has also captured smaller features and reflected more details of the surface than TLI.

Table 7.1: (1): 2D maps produced by Shepard and TLI at various network densities.

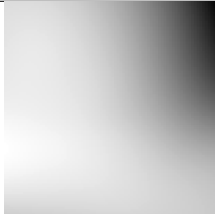
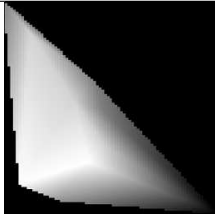
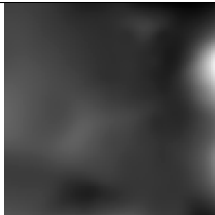
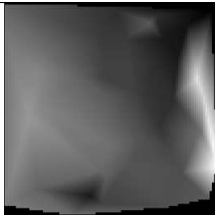
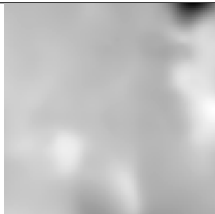
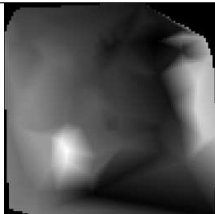
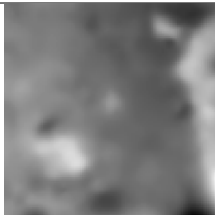
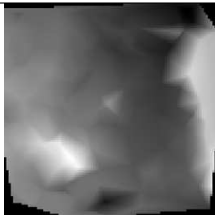
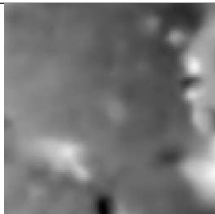
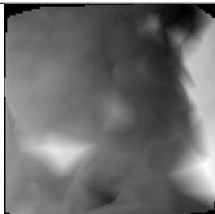
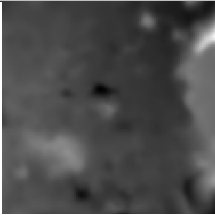
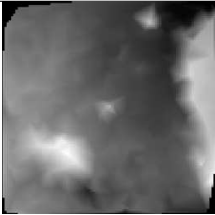
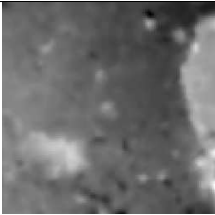
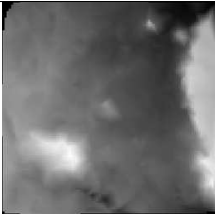
Network density	Shepard	TLI
10		
50		
100		
200		
300		

Table 7.2: (2): 2D maps produced by Shepard and TLI at various network densities.

Network density	Shepard	TLI
500		
1000		

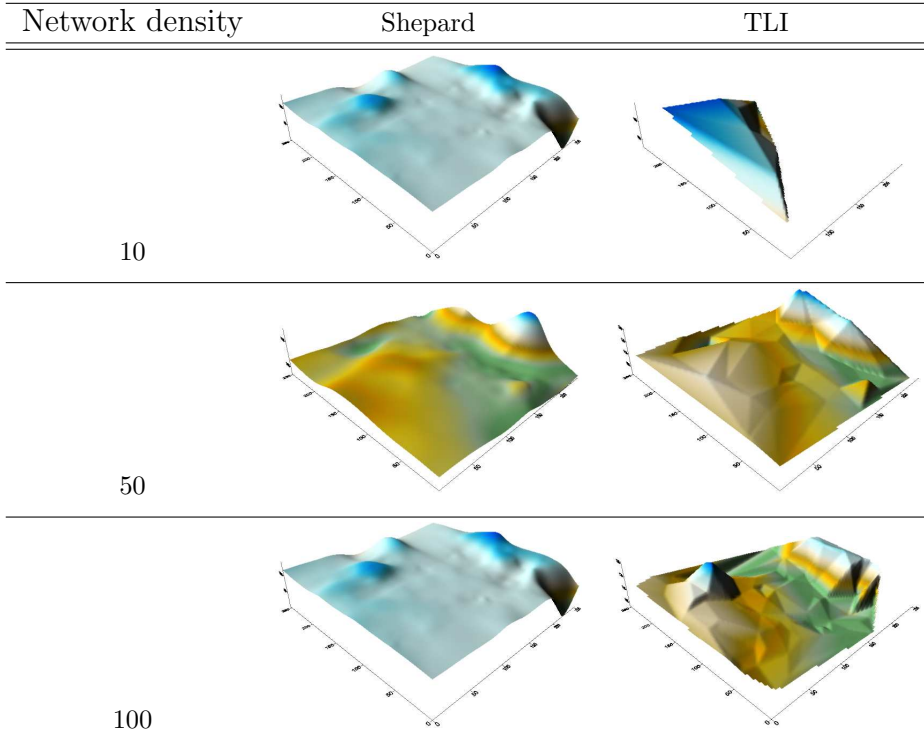
Experiment 2: Interpolation Local Behaviour

Aim: In this set of experiments the local performance of Shepard and TLI interpolation algorithms is to be evaluated through application of image processing approaches.

Procedure: Peak profiling and statistical measures (mean, skew, and kurtosis) are used to quantitatively characterise and compare local features extraction capabilities of both interpolation algorithms at various network densities. The highest peak in the Grand Canyon height data, labelled in Figure 7.3, was selected as the local feature that is quantised from maps produced by each interpolation algorithm at various network densities.

Results and discussion: Figures 7.4 to 7.10 show the profiling results of the selected peak from the original map and from maps produced by the Shepard and TLI interpolation algorithms. It is observed from the figures that Shepard interpolation appears to be more visually plausible and has always rendered a smoother surface than TLI. Shepard had consistently equal or better results than TLI because it adhered more to the original

Table 7.3: (1): 3D maps produced by Shepard and TLI at various network densities.



height range of the data. It can be seen that the surface generated by Shepard method is less “pointy”, i.e., its curve at the peaks of the observation points is lower than that obtained using TLI interpolation functions. Besides producing surfaces that are further from the actual surface, TLI is less smooth than Shepard. This is because that TLI surface passes through all points whose values are known. As explained in Section 4.2, there is an inherent uncertainty in the sample values, and so it is not strictly essential to interpolate the sample point values. Shepard curves maintain the local shape properties of the nodal functions, however, there is a mild decrease in a point’s influence as it gets farther from the prediction location. While in TLI curves, all locations within the relevant triangle get the same weight regardless of how far they are from the prediction location. In local Shep-

Table 7.4: (2): 3D maps produced by Shepard and TLI at various network densities.

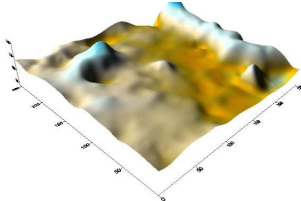
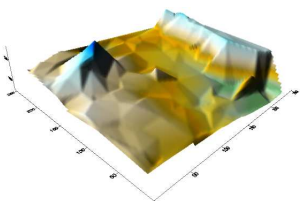
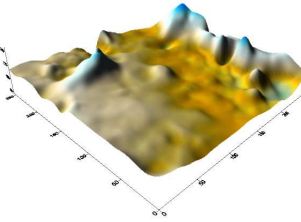
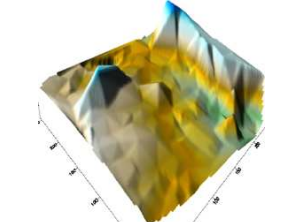
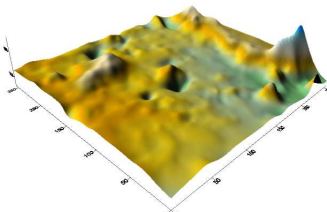
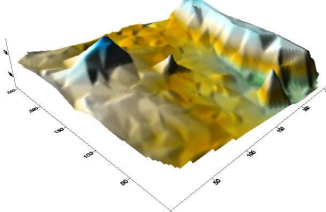
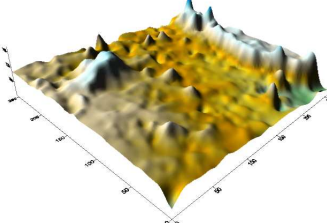
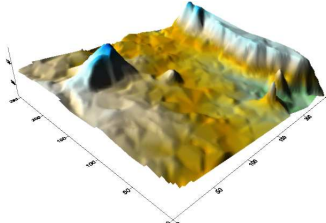
Network density	Shepard	TLI
200		
300		
500		
1000		

Table 7.5: Peak profiling statistical measures: Mean, Skewness, and Kurtosis

Network density	Skewness		Kurtosis	
	Shepard	TLI	Shepard	TLI
10	-0.340	-0.129	2.072	2.085
50	-0.314	-0.105	2.088	1.782
100	-1.698	-0.610	5.372	2.086
200	-1.065	-1.069	2.862	3.380
300	-1.635	-1.613	4.669	4.486
500	-1.117	-1.186	4.211	3.465
1000	-1.249	-0.964	4.211	3.092

ard interpolation, the enforced restriction of support set to sample points within the neighbourhood reduced the effect of distant points and produced a final surface that is visually much closer to the original for some features. At low network densities (10 and 50) Shepard outperforms TLI as it yields a surface which is much more representative of the original surface than that yielded by TLI. This is because TLI requires a medium-to-large number of data points to generate acceptable results. With a highly variable surface such as this, 50 data points are insufficient for TLI to re-create the source data, despite the relatively small size of the region in question. At all network densities, TLI suffer from edge effects because data sets that contain sparse areas result in distinct triangular facets on a surface plot or contour map. At slightly higher network densities (100 and 200), TLI was less representative of the original data range than Shepard because it tends to capture broad regional trends in the surface. TLI does not provide the ‘flatness’ on the edges we would hope for. As the network density increases, both interpolation algorithms give almost equal results with better performance from the Shepard algorithm on the basis of adherence to the original surface.

Table 7.5 presents a summary of statistics for peak profiling results at various network densities using Shepard and TLI interpolated maps. The

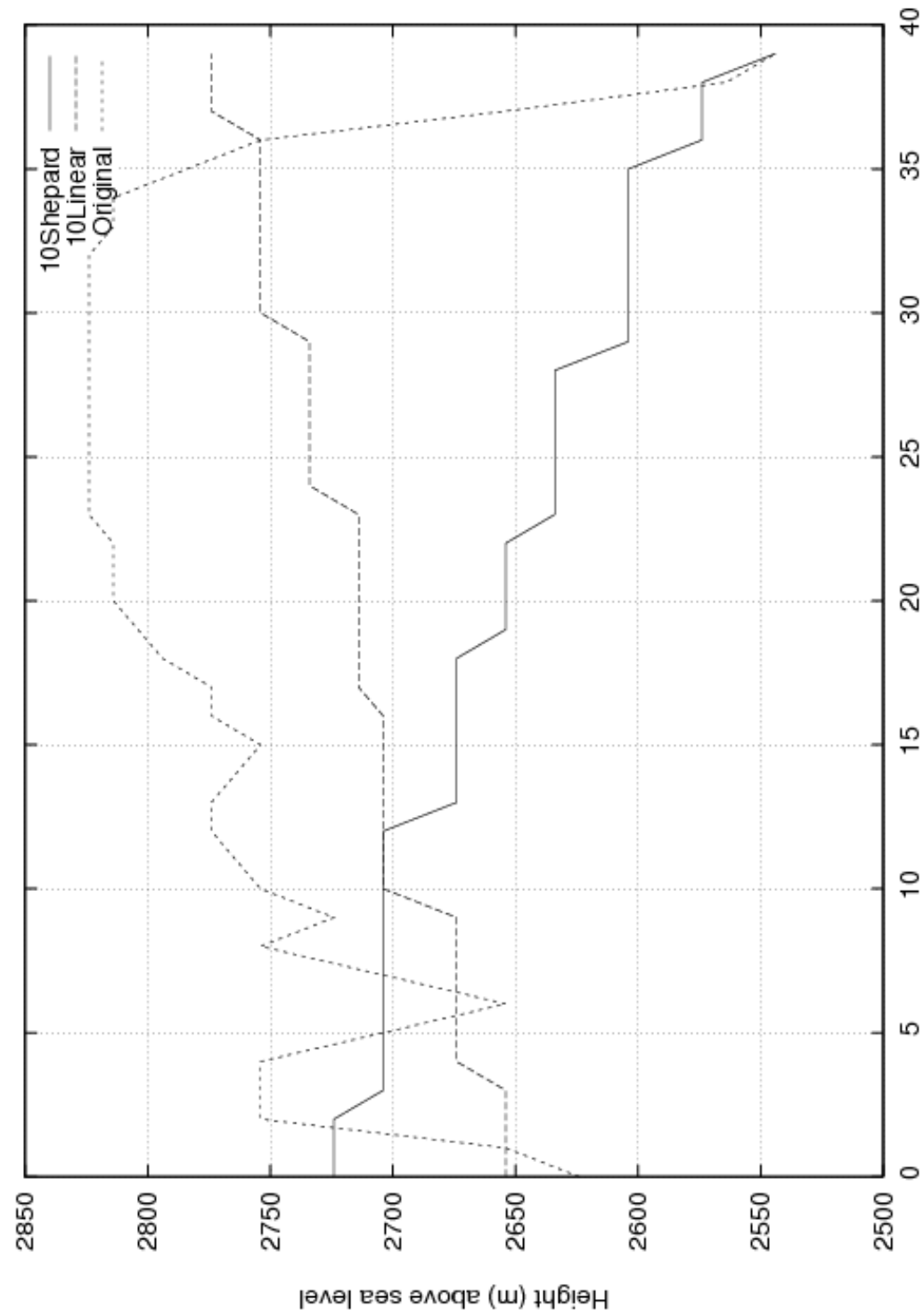


Figure 7.4: Peak profiling with 10 nodes network density

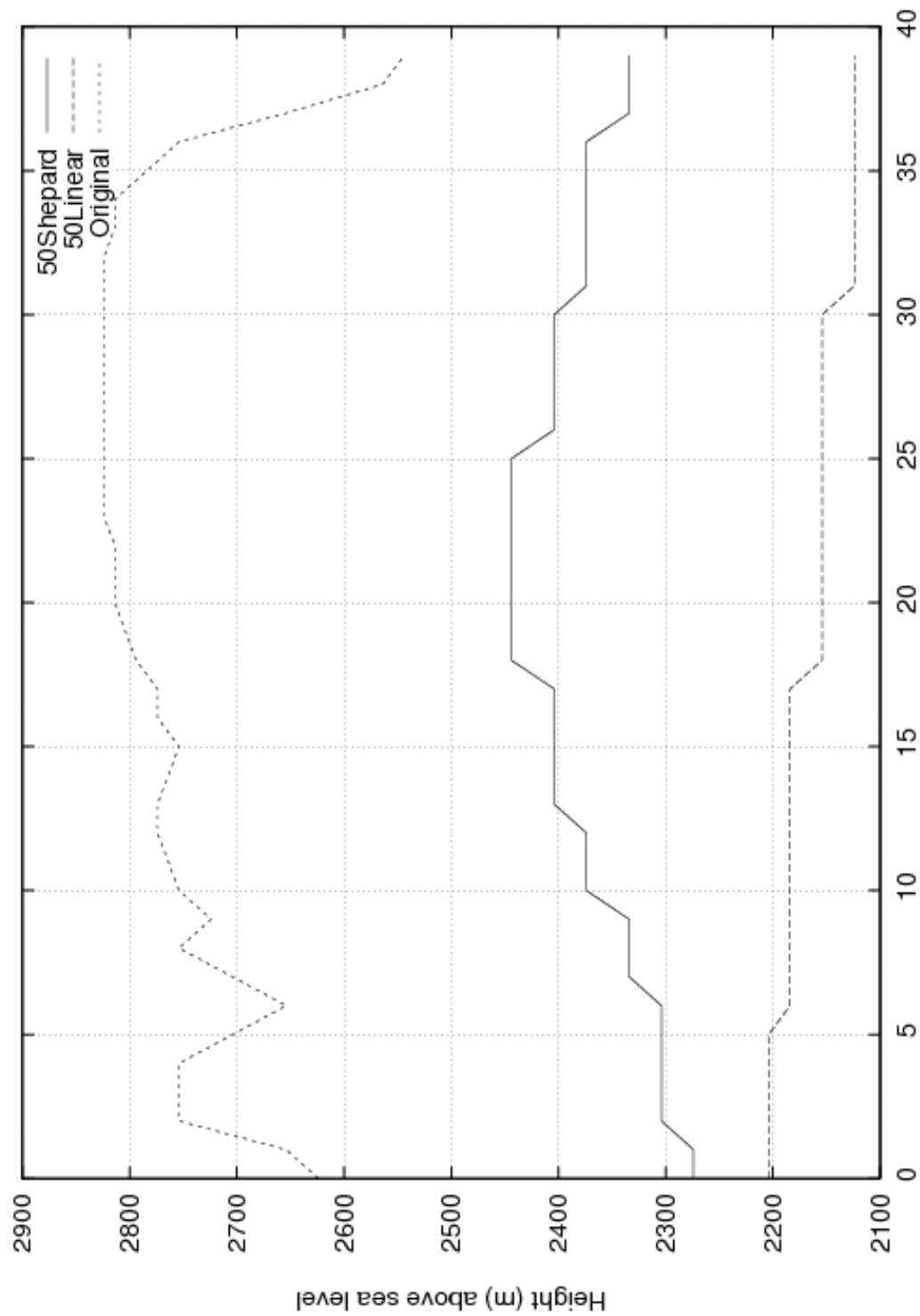


Figure 7.5: Peak profiling with 50 nodes network density

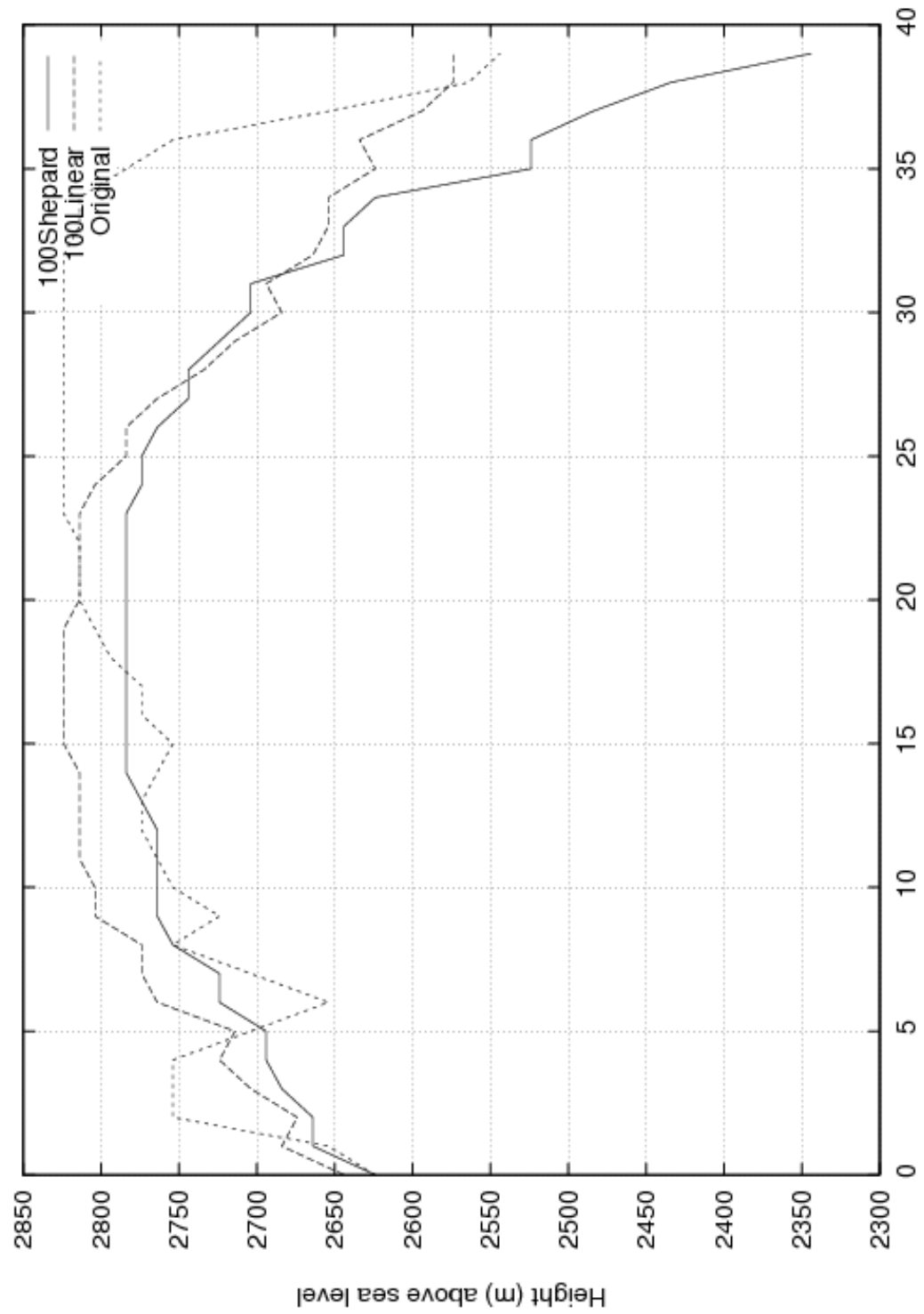


Figure 7.6: Peak profiling with 100 nodes network density

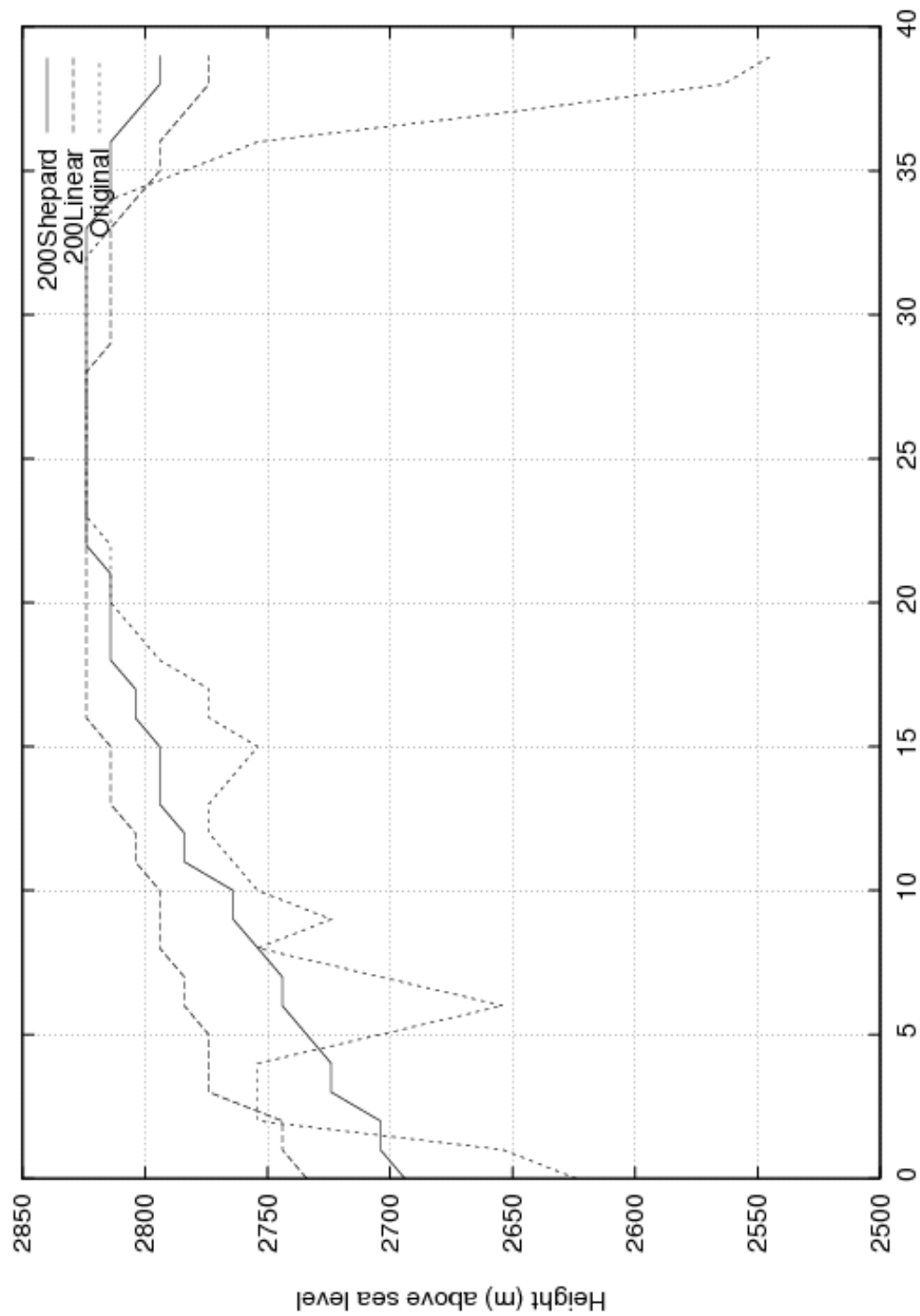


Figure 7.7: Peak profiling with 200 nodes network density

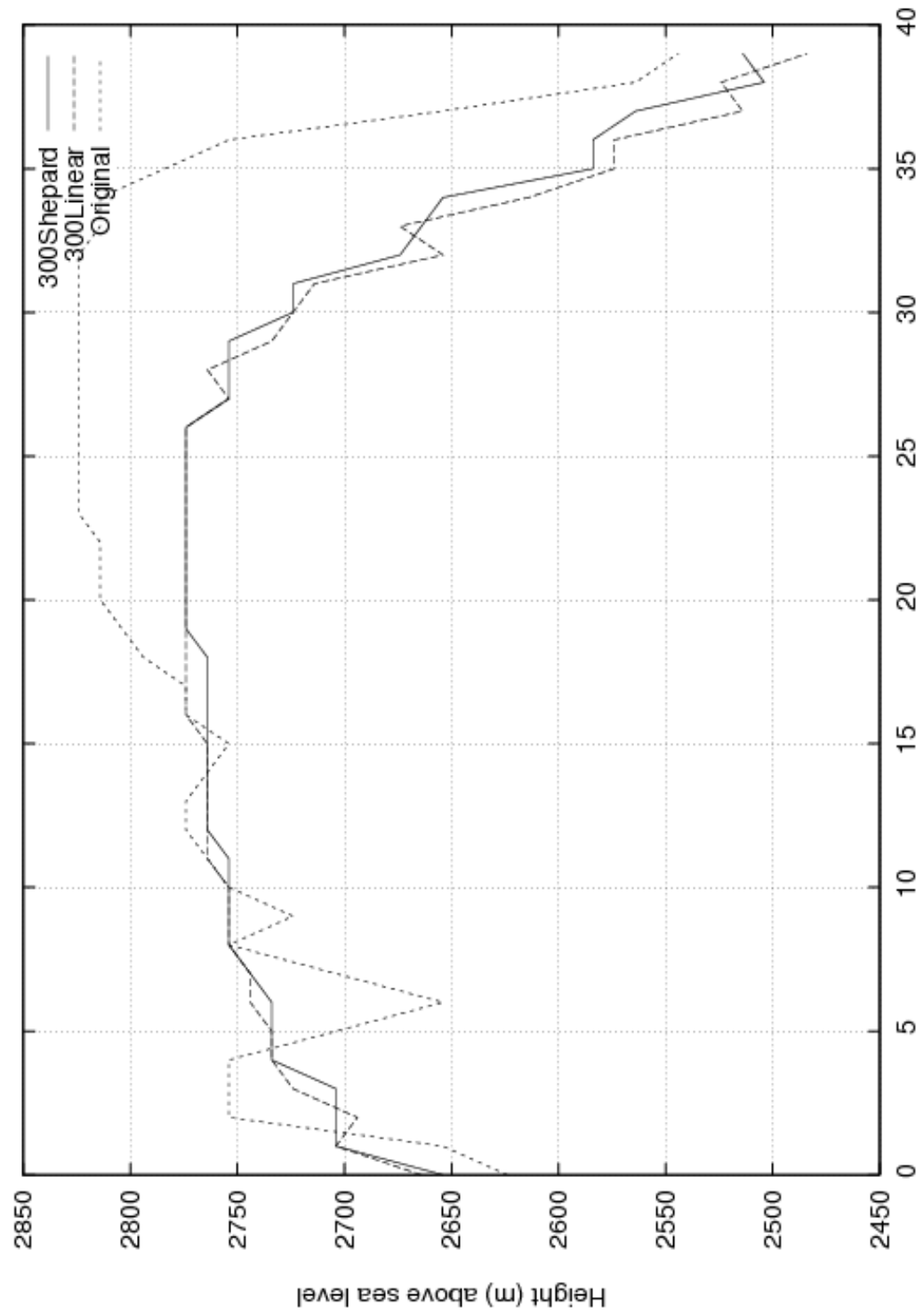


Figure 7.8: Peak profiling with 300 nodes network density

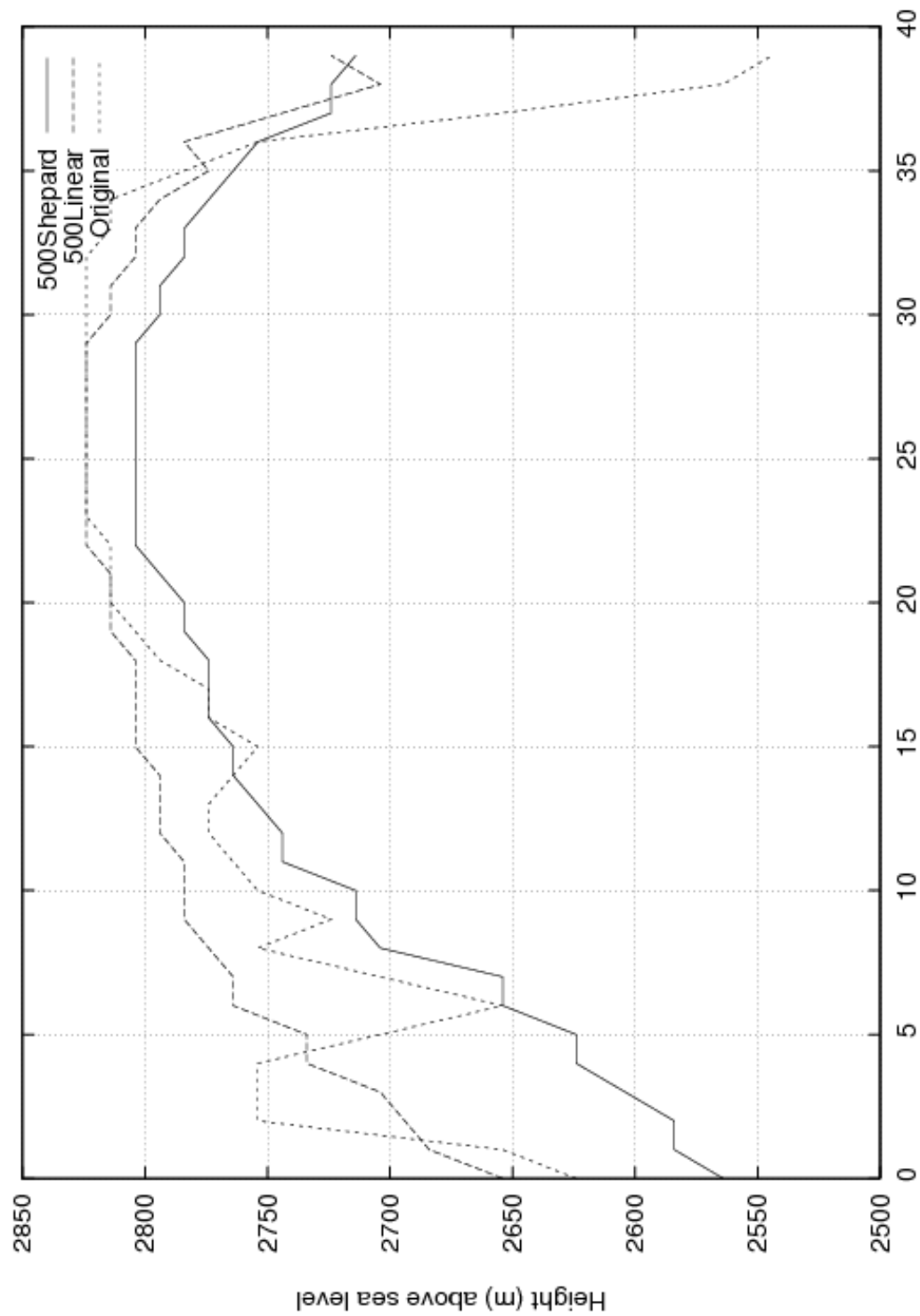


Figure 7.9: Peak profiling with 500 nodes network density

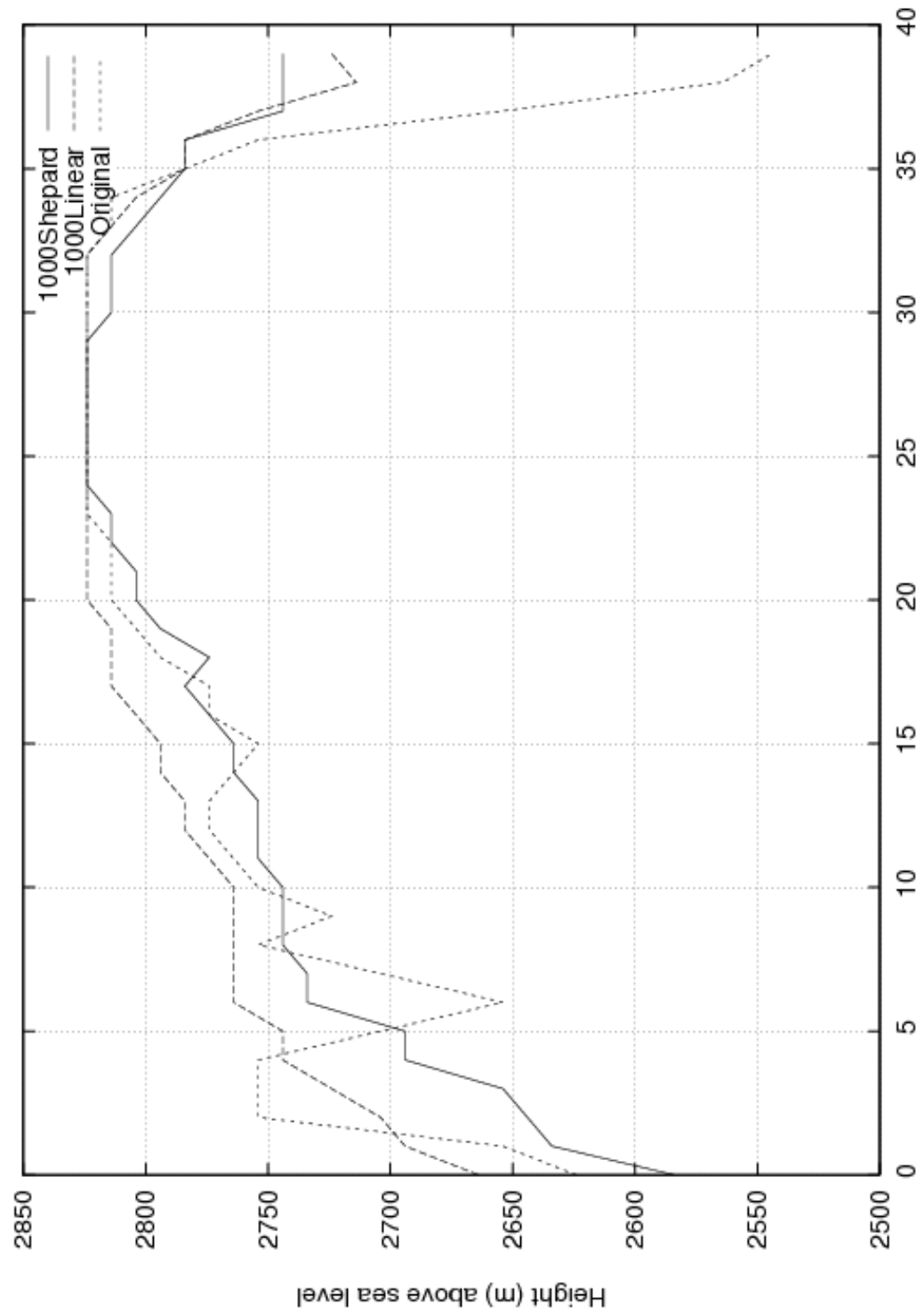


Figure 7.10: Peak profiling with 1000 nodes network density

skewness and kurtosis values measured from the original map are -1.419 and 4.423 respectively. The values recorded in table 7.5 shows that as the network density increases, the quality of the produced maps increase. The skewness measurements in table 7.5 confirm the results found in the previous experiment that Shepard interpolation produced a more accurate presentation of the interpolated surface at smaller data sets. However, with bigger data sets (200 and 300) the peak deviation difference of Shepard and TLI interpolated surfaces from the original surface is minimised. Looking at the kurtosis measures, Shepard gives more accurate results of how peaked a distribution is. This success of Shepard was due to the use of a subset of the observation points which is more related to the interpolation location and ignores the effect of distant points.

Conclusion: The results of these experiments, summarised in table 7.5, showed that Shepard interpolation was more capable of extracting local features of the interpolated terrain than TLI. This result makes the Shepard interpolation method more suitable for implementing the localised mapping service in large wireless sensor networks.

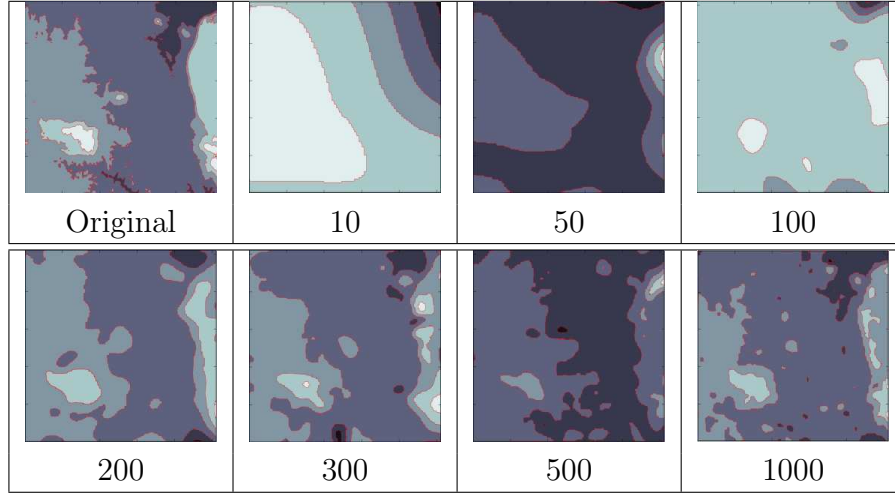
Experiment 3: Acceptable Level of Data Presentation

Aim: In this experiment the question of what is an acceptable level of data presentation needed for a particular application was investigated.

Procedure: The required accuracy level of interpolated maps may vary significantly depending on the specific application. The contour maps presentation style was chosen as an application to determine the network density required to reflect some terrain characteristics with particular levels of accuracy and details. In this set of experiments the quality of the constructed contour maps is to be evaluated. Specifically, the effect of the network density on the quality of the contour maps is to be studied.

Contour maps are a variety of maps characterised by large-scale detail and quantitative representation of terrain, usually using contour lines (Li

Table 7.6: Contour maps drawn on maps produced by Shepard interpolation



et al., 2005; Andrews, 1982). Contour lines connect a series of points of equal elevation and are an efficient way of representing a phenomenon on the terrain. As in Experiment 2, the contour map presentation of the highest peak in the monitored area will be studied.

We identify three factors that affect the data presentation quality: the interpolation algorithm, data characteristics, and network density. These three factors were studied in previous sections, however the representation of various features shown on the contour map is highly affected by the network density due to shorter line segments connecting two data points. Thus, we perform further investigation on the network density factor. We restrict the data representation quality experiments to Shepard's algorithm because Experiment 1 and Experiment 2 proved that Shepard is more suitable for spatial data interpolation at lower times and processing complexities than TLI.

Results and discussion: Table 7.6, shows a number of contour maps overlaying the height map generated using the Shepard interpolation algorithm at various network densities. By comparing contour maps constructed us-

ing low (10, 50, and 100), medium (200 and 300), and high (500 and 100) network densities, it is noticed that the reconstructed maps are very similar to the original one. The lowest network density at which the selected peak was successfully captured is 100, however it did not precisely identify the size of the peak. At 200 nodes network density both the size and the height of the peak were represented correctly on the contour map. With higher network densities, contour maps exactness increased rapidly and the similarity between the contour maps generated using 200, 300, 500 and 1000 is insignificant. Thus a network density of 200 is enough to give an acceptable presentation of that desired feature.

Conclusion: From the contour map and the peak profiling results, it can be seen that most of the topographic variations of the terrain were represented with accuracy levels enough to supply information on the topography of the land surface at a 200 nodes network density. In current real-life wireless sensor network deployments, this network density is achievable which proves the appropriateness and efficiency of Shepard interpolation when applied in wireless sensor networks.

The results from experiments 1, 2, and 3 show that local Shepard interpolation is a feasible approach for spatial data interpolation. The Shepard method is also suitable from an application point of view. It provides a more indicative image on the spatial relationship between geographical blocks and reveals some faults which other methods trialed can not reveal.

7.4 Chapter Summary

This chapter has discussed the proposed new network map generation service. We prove by empirical analysis that Shepard is a suitable algorithm for the distributed wireless sensor networks data extraction and visualisation mapping service. This algorithm was found to be an attractive solution due to its wide use in the literature. Furthermore, Shepard in-

terpolation is intuitively understandable and provides a large variety of possible customisations to suit particular purposes, e.g. relaxation of the interpolation condition for noisy data sets (Ruprecht et al., 1995). Also, Shepard can be easily modified to incorporate different external conditions that might have an impact on the interpolation results, such as barriers. Shepard's method is simple to implement with fast computation and modelling time (Krivoruchko, 2004; Lazzaro and Montefusco, 2002) and an easy generalisation to more than two independent variables. This method can be localised, which is an advantage for large and frequently changing data sets, making it suitable for sensor network applications. Local map generation by Shepard function reduces data communication across the network and evades the computation of the complete network map when one or more observations are changed. Finally, there are few parameter decisions and it makes only one assumption which gives it the advantage over other interpolation methods (Krivoruchko, 2004).

Chapter 8

Distributed Map Generation

The distributed construction of accurate and efficient visualisations of sense data gathered from wireless sensor networks is an open problem. This chapter presents a novel solution in which groups of network nodes cooperate to produce local maps which are cached and merged at a sink node, producing a low-resolution map of the global network. The sink node receives periodic map updates from each cluster head used to refine an up-to-date global map. The global map gives a low cost interface used to target queries for generating detailed maps from a subset of the sensors in the network. Different map resolutions are adequate for different management applications to perform at a desired level. In an energy constrained network, it is preferable to allow the application to control the level of detail and topology coverage of a generated map.

Solving global network data mapping problems through localised and distributed computation is achieved by abstracting and simplifying the routing and the map generation services and turning them into services of the network rather than being the result of coordinating the services of individual nodes. This system changes the mechanism of use of certain capabilities in wireless sensor network; the interpolation service is used by the sensing nodes to deal with mapping data and a cluster-based routing algorithm in

which the cluster heads are selected as the caches for local maps.

8.1 Introduction

Modern advances in measurement and instrumentation have required increasingly sophisticated visual representations, to ensure that scientists can quickly and accurately interpret increasingly complex data. Most recently, wireless sensor networks have emerged as a technology which can provide high fidelity, multi-modal sense data over large geographic areas and long periods of time. Traditionally, such applications as mapping would be performed at a single host computer, situated outside the wireless sensor network. In this case, the sensor network is simply a data gathering tool, with all of the information processing being centralised within the computer. Increasingly wireless sensor networks are being seen as a more open and re-targetable resource, possibly with multiple or mobile users and consequently multiple sinks. In this situation it is necessary for information processing to occur within the network, rather than in an external computer.

Graphics algorithms are often computationally expensive which can cause problems for low-power, long-term deployments of sensor networks. Enough data must be cached in the network to generate maps, although memory may be a scarce resource. Nodes must cooperate to interpolate between data gathered at different points in the network, but radio communication may need to be minimised for energy efficiency.

This chapter presents a novel solution to these problems, in which groups of network nodes cooperate to produce local maps which are cached and merged at the sink node defined for the mapping transaction, producing a low-resolution map of the global network. The sink node receives periodic map updates from each cluster head used to refine an up-to-date global map. The global map gives a low cost interface used to target queries for generating detailed maps from a subset of the sensors in the network. The

key contribution of this work is to combine the mapping and routing services of wireless sensor networks, by utilising a cluster-based routing algorithm and selecting cluster heads as the caches for local maps.

For many mapping applications, a complete, high resolution map is not needed. Obtaining the map of an area of interest often suffices. The approach presented here allows the user to obtain a complete or partial map of the environment at the desired level of fidelity. A complete map can be obtained by generating local maps at each cluster; the integration of these partial solutions solves the global problem. Indeed, the complete map is calculated by merging sub-maps produced locally at clusters.

The cluster-based technique presented here is a quasi-distributed solution, as individual clusters of nodes behave as if they were a centralised network. This is an improvement on centralised mapping services which require all relevant data to be transmitted to the sink node to produce a map, resulting in serious communication bottlenecks.

Hierarchical data distribution is acknowledged as an effective approach to reducing the communication in the network and load balance energy consumption (Chang et al., 2006; Ganesan et al., 2003; Chan et al., 2006). The experimental work presented here uses the MuMHR routing algorithm (see Chapter 6) that handles various topography types such as forests, or building interiors with obstacles of any shape, without implementation changes.

The method proposed here is automatic and does not depend on a specific topology. Maps are successfully handled using no domain specific knowledge. This decentralised approach works very well in domains with a dynamically changing environment. The approach potentially enables the network to adapt to changes in the monitored environment locally (and independently of the other network clusters) which would make the scheme scalable. If desired, more sophisticated application-specific algorithms may be adopted as “plug-ins” for increased performance.

In this chapter a distributed in-network mapping service which utilises a

range of network topologies, routing protocols and interpolation algorithms is proposed. Section 8.4 gives an experimental evaluation of the technique, carried out in simulation.

8.2 Distributed Map Construction

The hierarchical organisation of the data storage, extraction, and processing scheme can be exploited to search efficiently for patterns across the network. This can be done by first extracting and examining a low-resolution map at the sink. Then this map can be used to obtain more highly accurate maps for sub-regions of the network efficiently by drilling down the routing hierarchy by eliminating clusters whose maps are not significant to answering a query. In dynamically changing environments, when local changes occur on topological elements, such as fire causing a wall to collapse, the map produced from the modified cluster is recomputed locally and the rest of the map is left unchanged. The routing hierarchy can have any number of levels, making it scalable for large problem spaces. When the problem space is large, a larger number of levels can be the solution for reducing the map production effort, for the cost of more localised storage and pre-processing time.

Distributed Mapping Requirements

The highly data-driven nature of sensor networks and their limited resources impose many requirements on data storage and processing infrastructure. A fully centralised data collection, storage, and extraction are infeasible for many reasons including: the high energy cost involved in transmitting the data to the sink; it introduces a single point of failure; performance bottlenecks especially around the sink; significant redundancy of sense data; a big portion of the returned data is not useful; among others. However, many queries of the sensor network are of a spatio-temporal nature that

requires centralised data collection. Mining for characteristics of such nature where local processing is not enough should be feasible. Different map resolutions are adequate for different management applications to perform at a desired level. In these cases, it is an unjustified waste of resources to retrieve the entire topology of large-scale networks especially as sensor nodes are energy constrained. Furthermore, a sensor network user will regularly require an abstract global view of the large sensor data. A global map gives a visual interface which allows users to drill down into areas that appear to contain noticeable anomalies without requiring a pre-defined notion of what makes up such anomalies. For instance, path-finding problem frequently put forward in disaster recovery scenarios such as forest fires. At slightly higher level views, the details of the monitored environment fade, facilitating planning at a higher level. Then the responders can zoom-in over the identified location of the fire to plan access to the site.

Distributed Mapping Service Architecture

The distributed mapping service is made up of four modules: *Application*, *Interpolation*, *In-network Processing* and *Routing*. Figure 8.2 shows the mapping service architecture and interaction between its four modules.

The Routing module is an essential module responsible for data communication. In this scheme, MuMHR, discussed in Chapter 6 is the routing protocol used. The defined routing procedure builds the hierarchy and establishes the path between sensing nodes and their respective cluster head to enable data transmissions.

The role of the In-network Processing module is to process raw data received from various cluster head nodes in the network. It applies filtering on all the received data to reduce redundancy resulting from overlapping cluster coverage. Moreover, the In-network Processing module manages incremental update messages and merges them into single transactions. Also, it defines two interfaces for the Interpolation and Application modules through

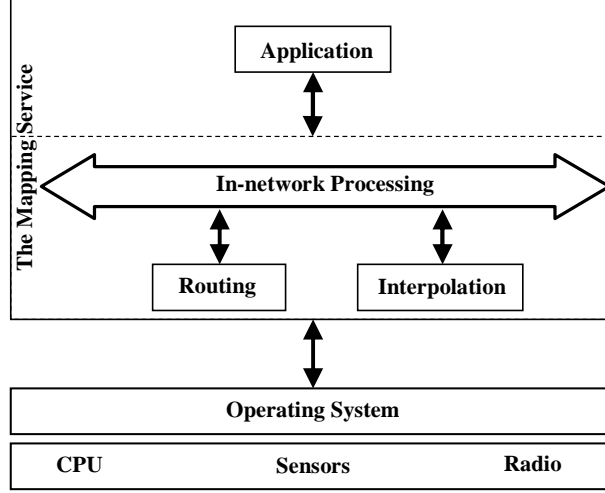


Figure 8.1: Architecture of the distributed in-network mapping service.

which it provides access to the cached data in a suitable format.

The Interpolation module represents the map generation service. All mapping applications use the Interpolation module as a building block to generate maps. The Interpolation module provides access to the In-network Processing module to obtain the available mapping or update data. In this scheme, we use Shepard interpolation (Shepard, 1968) discussed in Chapter 7.

Finally, the Application module contains the user defined applications such as path-finding or isopleth maps. The application module also has direct access to the In-network Processing module to get raw data if required.

An Algorithm for Distributed Map Construction

The system proposed here responds to the requirements identified in Section 8.2 and integrates them into a distributed, in-network mapping service. It includes a hierarchical, multi-resolution storage and a distributed processing to data extraction paradigms. Scalable, low-resolution, and load-

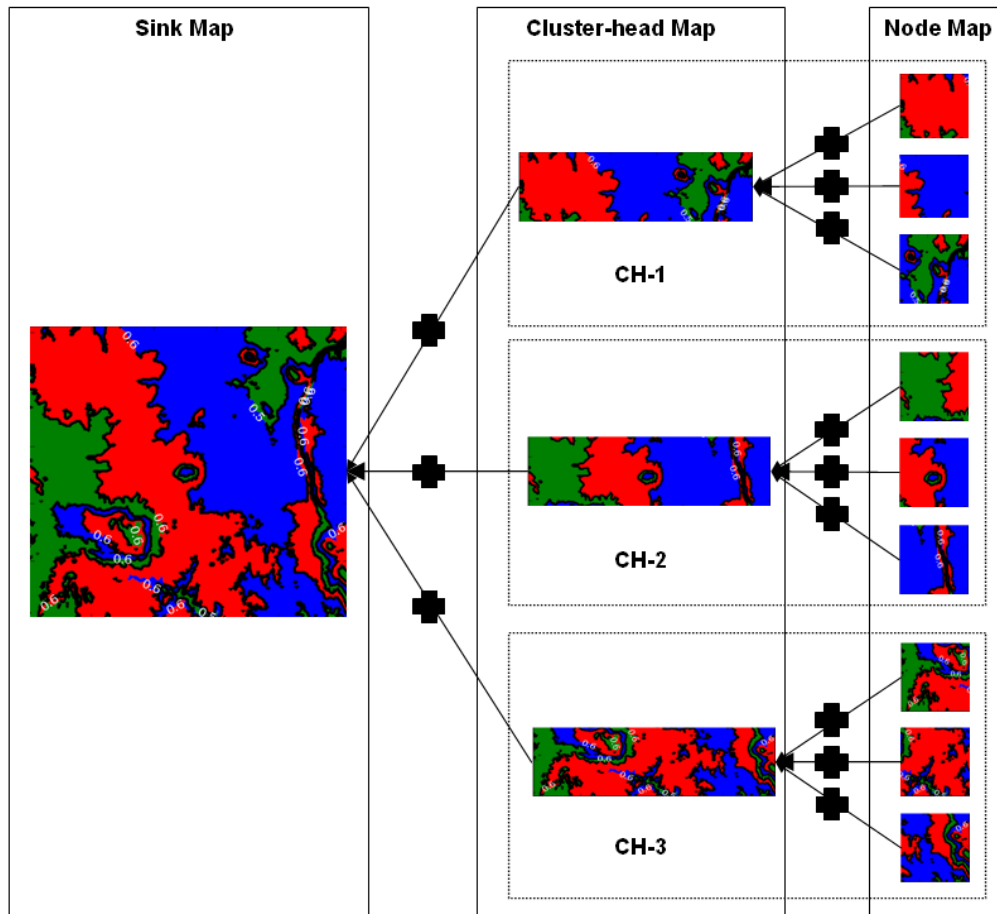


Figure 8.2: The problem of generating a complete network map is solved by merging sub-maps produced and maintained locally at clusters-head nodes. The filled Plus shape represents an incremental map update message.

balanced data extraction is sought as a solution to fulfil these requirements. The sink generates a low cost, low-resolution map for the network region before deciding to get more detailed and more expensive detailed data sets. At a lower hierarchical level, cluster head nodes maintain high-resolution maps from a larger number of nodes to obtain an accurate image from a smaller region. Figure 8.2 illustrates how the problem of generating a complete network map is solved by merging sub-maps produced and maintained locally at cluster head nodes. It shows how local cluster head nodes maintain a high-resolution accurate map of the regions they manage; these maps are merged at the sink to generate a complete map of the network that contains lower-level of details from larger region.

The major steps of the distributed mapping service are as follows:

1. Cluster head node collects data from its vicinity
2. Cluster head node computes its partial map and forwards it to the sink
3. The sink computes complete network map
4. Dynamically update the partial and complete network maps

Cluster head collects data from its vicinity

After the paths to the cluster head are established and every node receives a TDMA schedule, sensing nodes start data transmission to their respective cluster head in their assigned time slot. The data is sent to the cluster head in the form of a tuple (x, v, t) where x is the 2D Cartesian coordinates of the sensing node, v is the value of the measured parameter(s), and t is a time-stamp denoting the time at which the data packet was created.

Algorithm 1 describes the behaviour of sensor nodes where s_v is a set containing the last read value of each sensor attached to that node. A sensor node will request a fresh reading, v_i , from all sensors, N , attached to

Algorithm 1 A description of sensing nodes behaviour.

PROCEDURE CollectData(s_v, T)

```

 $b = []$ 
FOR  $i = 1 \dots N$  DO
  request  $v_i$ 
  IF  $T_i == 0$  THEN
    createLocalAccuracyModel()
  END IF
  IF  $s_{v_i} == 0$  THEN
     $s_{v_i} \leftarrow v_i$ 
     $b.append((i, s_{v_i}))$ 
  ELSE IF  $|v_i - s_{v_i}| > T$  THEN
     $s_{v_i} \leftarrow v_i$ 
     $b.append((i, s_{v_i}, t))$ 
  END IF
  IF  $T_i == 0$  THEN
    createLocalAccuracyModel()
  END IF
sensor.send( $(x, b)$ )
 $b = []$ 

```

it. If the node was in the initial data collection phase it computes the local accuracy model and stores a copy of all its sensors readings. In latter data transmission phases, sensor nodes classify each new reading according to the local accuracy model, T , and then send their cluster head only readings that are classified as significant changes.

Though attempting to achieve a desired global objective, the sensing nodes interact with each other only within a restricted neighbourhood, defined using the interpolation service. A sensing node only buffers data packets for neighbouring nodes that have a significant effect on its local map. For instance, Shepard interpolation defines an arbitrary number of nodes, N , within a particular radius from the node, R , as the set of nodes which has effect on the local map. For more information about how N and R are calculated see (Shepard, 1968).

Algorithm 2 A description of cluster heads behaviour.

PROCEDURE ClusterHeadTask()

WHEN receive sensor update

$c++$

$P_{map}.update()$

IF $c == m$ **THEN**

$sd = createSummaryData()$

END IF

 sensor.send(sd)

The cluster heads continuously record and aggregate each nodes data. To avoid sending large amounts of data across the network, various cluster head nodes will collect the data from their area and create summary data sets as defined by the In-network Processing model. The size of a summary data set is determined with the accuracy level required by the complete network map. These summary messages are then transmitted to the sink.

Cluster head computes its partial map and forwards it to the sink

The cluster head uses the received data from nodes within the cluster to build an accurate map of its vicinity. To reduce the size of the transmitted data set, the cluster head applies compression techniques defined in the In-network Processing module to eliminate redundancy resulting from overlap of nodes coverage area. Given an up-to-date data set, the cluster head can easily generate a partial map using the map generation service. The compression process is important as it makes the interpolation step faster. The cluster head caches two data representations: point data and graphical map. The former is used to build the local graphical map and forwarded to the sink to be combined with data received from other clusters to build the complete network map. While the latter is used to respond quickly to user queries about events or parameters to be found within the cluster area.

Algorithm 3 Incremental update of local and global maps.

```

IF node is a sensing node THEN
    create a local accuracy model
    classify each new reading according to the local accuracy model
    IF local accuracy deviates THEN
        update the local map
        forward the new reading to its cluster head
    END IF
END IF
IF node is a cluster head THEN
    WHEN receive update message
        update local map
        send the update message to the sink
    END IF
sink combines up-to-date local maps

```

Algorithm 2 describes the behaviour of a cluster head node where m is the number of members in that cluster and P_{map} is the partial map.

The sink computes complete network map

The sink fuses the partial map data received from all cluster head nodes into a complete map of the network. By applying the same interpolation step performed at the local cluster head nodes, the sink generates a complete map of the network. When requested by the user, this map can be refined by querying a more detailed map from the cluster head(s) managing the area of interest. A user may want to query detailed data sets from a group of sensors in the network. A query may be directed to the cluster head node nearest to the desired location.

Dynamically update the partial and complete maps

In many real-world applications of wireless sensor networks, the sensed modalities are mostly unchanged or slightly changed over time (Xue et al.,

Algorithm 4 Dynamic calculation of nodes local accuracy model.

- (1) cache data of neighbouring nodes
 - (2) use the map generation component to calculate sensor own reading using the neighbours data
 - (3) calculate the standard deviation form the estimated value from 2 and the actual reading
 - (4) repeat steps 1 to 3 for each of sense modality
-

2006). Therefore, the successive maps generated for these modalities are very similar and updates of the current map are enough and could save a considerable amount of energy. However, in large scale wireless sensor networks, map update data could be large in size. The standard incremental update method of partial and complete maps is large if compared to the bandwidth of a typical wireless communication channel. Map update size reduction is therefore of vital importance.

One method for map update size reduction is that each node uses the Interpolation module of the mapping service to build a local accuracy model (see Algorithm 4) which can be used by the node to base its decisions of reporting a change in its local map to its cluster head. The local accuracy model is defined such that a node is able to estimate its own reading using other observation points within some level of tolerance. This level of tolerance can be derived from the global cluster map accuracy. Each sensor node tracks its local accuracy model. When the accuracy level changes sufficiently, the sensor node saves the new calculated map using the most recent readings and it reports these changes to its corresponding cluster head. If several updates arrive at the cluster head about the same time, they are grouped into one update transaction before being sent to the sink. Algorithm 3 outlines the approach.

Table 8.1: Linux-class sensor node hardware platforms.

	MicroServer	Gumsense
CPU	RMI Alchemy au1550	MarvellPXA270 withXScale
Clock speed	400MHz	100 MHz to 600MHz
CPU power consumption	0.5W	72 μ W
Memory	128MB	128MB
Flash ROM	128MB	32MB

8.3 Hardware Limitations

The following experiment and the experimental work in Chapter 9 target sensor networks built from Linux-class devices that have higher storage and processing capabilities. The choice of less constrained hardware platform was for two reasons:

1. In-network distributed sense data mapping is desirable but introduces a considerable storage and computation complexity on sensing devices when considering current sensor node capabilities.
2. In-network visualisation has requirements typical of any non-trivial processing. For example, the MICA/MICA2 mote (xbow, 2007) microcontroller has no support for floating point arithmetic or integer multiplications.

The Gumsense (Martinez et al., 2009) and EmStar MicroServers (Girod et al., 2007*a,b*), amongst other Linux boxes are example hardware platforms that are available in the market and are capable of running the distributed mapping service in a large-scale wireless sensor network. Table 8.1 shows the specifications of these hardware platforms.

In an extreme situation, assume that there is a sensor node that store 500 observation points. Assuming mapping data are represented as triplets of



Figure 8.3: A mineral map derived from AVIRIS data.

32-bit floats, the data alone requires 3.9KB of memory. Let I_d be the number of instructions required to estimate the value at any interpolation location. Knowing the clock speed of the processors allows making a simple estimate of the execution time. Combined with the 600MHz clock speed, execution time to calculate a partial map is estimated at 1.4583s. What is defined as an acceptable execution time is dependent on the application requirements.

8.4 Experimental Comparison of Centralised and Hierarchical Mapping Services

Aim: In this experiment, the efficiency of the proposed distributed mapping service in terms of energy and the quality of the produced map is to be

studied by simulating the distributed mapping service execution on a real life data set.

Procedure: This algorithm has been implemented in Dingo wireless sensor networks simulator, which is an enhanced version of SenSorPlus used to implement the centralised mapping service. To present the results of generating high quality maps at a low energy cost and minimal network delay we have chosen the determination of isopleths on interpolated fields. Figure 8.3 shows a mineral map derived from AVIRIS data obtained over Cuprite, Nevada, in 1995 (McCabe, 1998). The satellite spatial resolution is about 17 meters pixel spacing and the scene is $4.5km$ wide and $4.5km$ high and north is up. In this experiment, a 2000 node sensor network was randomly distributed over the area described above. The sink node is located at the centre of the covered area. Isopleths has been generated from the data collected by the centralised and hierarchical mapping services. The results obtained here are compared to the results of the centralised mapping service approach proposed in Chapter 5.

Results and discussion: The maps generated by the centralised and hierarchical mapping services are shown in Figures 8.4 and 8.5 respectively. By comparing the interpolation results, it is observed that the interpolated maps are visually very similar to each other as well as with the original map. This demonstrates that the proposed distributed mapping service is no less accurate than a centralised service. Quantitatively, the difference between the high-resolution source data and the interpolated data is measured as the RMSD difference of the pixels. Using this measurement, there is just a 1.2% difference between the hierarchical and centralised interpolation algorithms. One particular side-effect of the hierarchical method can be beneficial. The enforced exclusion of distant points in local interpolation can produce a final map that is visually much closer to the original for some features. Examine Figures 8.9, 8.10 and 8.11. The influence of irreverent distant points in the centralised interpolation (Figure 8.10) causes less of a similarity with

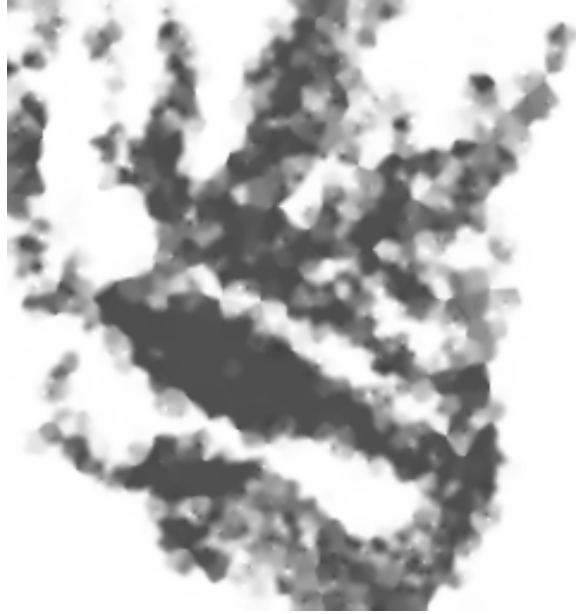


Figure 8.4: Interpolated field generated using a centralised mapping service running on 2000 simulated nodes, randomly distributed on the surface from Figure 8.3.

Figure 8.9 than the hierarchical method (Figure 8.11). Note the complete absence of the “loop” shapes under the main dark region in Figure 8.10.

The results of generating isopleths based on this data are shown in Figures 8.6 and 8.7. These contours were generated for a 116 units and a threshold of 1. The interpolated terrain generated by the two approaches is nearly identical and is similar to the real surface. However, the difference in the cost of generating the two contours is large. Figure 8.4 shows the difference between the number of messages exchanged using both approaches.

The comparison of the data shown in in Figure 8.4 captures the essence of the approach proposed here. The figure plots the mapping traffic against the number of nodes in the network. Remarkably, the number of exchanged mapping messages in the hierarchical mapping service is much smaller than that of the centralised version. This suggests that exploiting the routing



Figure 8.5: Interpolated field generated using a hierarchical mapping service running on 2000 simulated nodes, randomly distributed on the surface from Figure 8.3.

hierarchy results with substantial energy savings without any degradation in the quality of the produced map. Unlike the centralised mapping service, as the number of nodes in the network increases, the mapping traffic in the hierarchical service remained small which makes it a scalable solution for mapping large-scale wireless sensor network data. The distributed mapping service scales fairly well with respect to the network size and achieves these gains in the efficiency (energy and timeliness) without requiring any extra setup messaging.

Conclusion: The distributed mapping service produced no less accurate results than the centralised mapping service. However, the cost of producing the global network map from locally computed partial maps is much less than the cost incurred by the centralised mapping service.

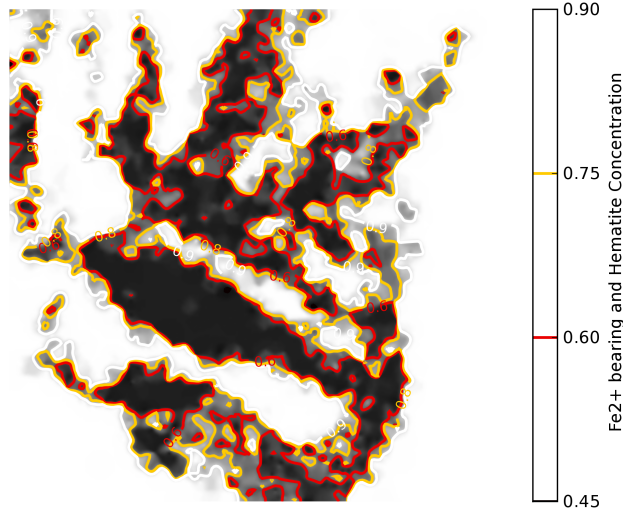


Figure 8.6: A contour at 116 units drawn on the interpolated field from Figure 8.4.

8.5 Chapter Summary

Visualisation of sense data gathered from networks of wireless sensors is a challenging problem in several regards. This chapter has presented a potential solution which addresses several issues, namely: energy efficiency, data storage and network organisation. The distributed mapping service is novel in that partial visualisations are computed and merged in local network clusters. This combination of routing and visualisation is the major contribution of this work. Results of this work, gained in simulation, indicate that significant savings in energy (from radio transmission) can be made using this distributed data extraction and visualisation mapping service; it is no less accurate than a centralised service; it requires only significant map updates to be transmitted from the sensor, rather than the entire raw data; and it allows each sensor to locally determine its behaviour and respond to

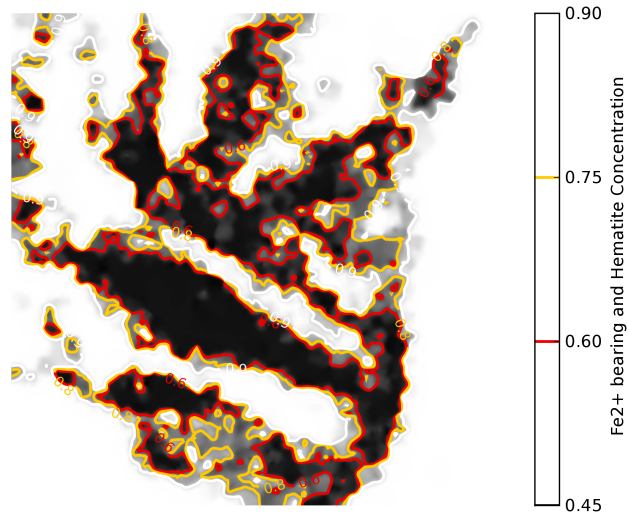


Figure 8.7: A contour at 116 units drawn on the interpolated field from Figure 8.5.

its environment, for example deciding when to recalculate its local map in response to changes in the input data.

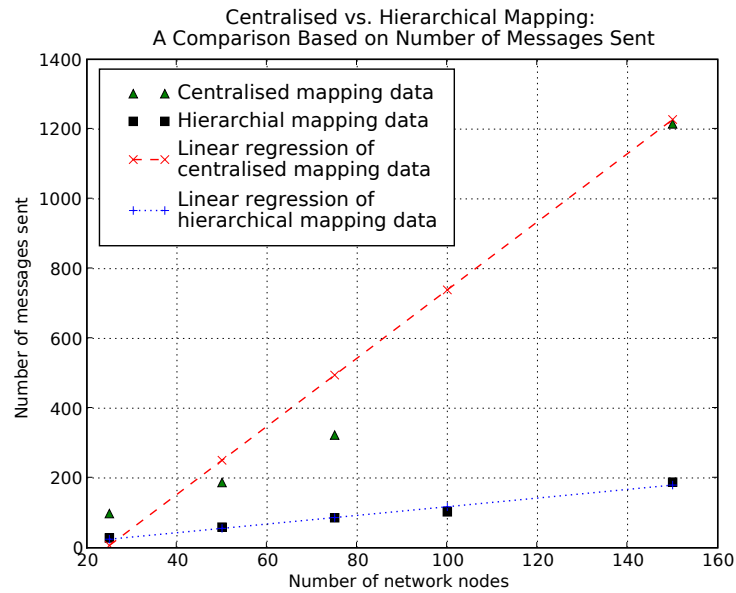


Figure 8.8: A comparison of centralised and hierarchical mapping based on number of messages sent



Figure 8.9: Detail section from Figure 8.3



Figure 8.10: Detail section from Figure 8.4, interpolated using the centralised method



Figure 8.11: Detail section from Figure 8.5, interpolated using the hierarchical method

Chapter 9

An Integrated Inductive-Deductive Framework for Sense Data Mapping

Wireless sensor networks have been useful in a variety of domains such as fire monitoring applications (Hefeeda and Bagheri, 2007; Hartung et al., 2006). They have an intimate interaction, via sensors, with the physical environment they operate within because wireless sensor networks are usually deployed to perform application specific tasks. These networks have an intrinsic interdependency with the environments in which they operate. It has been observed that such network applications are characterised by the tasks involved in a particular application. The part of the world with which an application is concerned is defined as that application's domain. This work advocates that an application domain of a wireless sensor network can serve as a supplement to analysis, interpretation, and visualisation methods and tools. The author believes it is critical to elevate the distributed mapping service capabilities to make use of the special characteristics of an

application domain. Particularly, this study upgrades the spatial interpolation method capabilities to interpolate an arbitrary number of variates by utilising the information given by the application domain to produce better interpolation results, i.e. the application domain is used to dynamically minimise the interpolation predictive error.

This chapter proposes an adaptive Multi-Dimensional Application Domain driven (M-DAD) mapping framework which is suitable for mapping an arbitrary number of dimensions and is capable of utilising the relations between different dimensions as well as other parameters of the application domain to improve the mapping performance. M-DAD is capable of dealing with dynamically changing application domain conditions. It starts with an initial user defined model that is maintained and updated throughout the network lifetime. The experimental results in Section 9.5 demonstrate that this M-DAD mapping framework performs as well or better than the distributed mapping service without its capabilities.

9.1 Introduction

Wireless sensor networks revolutionise the paradigm of collecting and processing information in diverse environments. The real worth of the enormous amount of data generated by the wireless sensor network lies not only in easy data access, but also in the additional possibility of extracting as well as presenting the implicit information contained in the data. Recently, such networks are being deployed for an increasingly diverse set of applications each with different characteristics and environmental constraints. As a consequence, scientists from different research fields have begun to realise the importance of identifying and understanding the characteristics and special deployment needs of different application domains. For instance, (Henninger et al., 1995) proposed methods for sharing knowledge about application domains by creating a repository of project experiences.

Such efforts resulted in the Public Domain Knowledge Bank also known as *PDKB* (PDKB, 2008) which is an artificial intelligence knowledge bank of commonsense rules and facts.

In many wireless sensor network deployments, the network owners have some knowledge about the monitored environment characteristics in which the target system operates. For example, in forest fire applications (Hefeeda and Bagheri, 2007; Hartung et al., 2006), information about the forest topography can be obtained from the Geographic Information System (GIS) or satellites maps. Another example is the environment monitoring application where information about the environment can be obtained from weather stations or simply knowing the current meteorological season.

A domain model carries knowledge of an application domain. It is a conceptual model of a system which describes the various real world entities involved in that system and relationships between them. The domain model provides a structural view of the system which we suggest using to complement the information gained from analysing data gathered by a wireless sensor network. In this chapter, we argue that by using additional knowledge available about the monitored environment, the distributed mapping service has a greater potential for giving more meaning to the collected data. The spirit of the approach presented here is to use a domain model to draw more meaning from the sensor network data using the structural view given by the domain model. The logical integration of a domain model and sensory data from multiple heterogeneous sensory sources can be effectively used to explain past observations as well as to predict future observations. It exploits the local semantics from the environment of each sensor. For example, if a motion sensor is located in a position where people are likely to walk, the local semantics enable the system to use “passage” statistical models, as opposed to a motion sensor covering an office space where people are usually sitting. It also takes advantage of human guidance and information from other available sources, e.g. satellites. Furthermore, it maintains

the overall coherence of reasoning about the gathered data and helps to estimate the degree of confidence using probabilistic domain models. The use of knowledge made available by the domain model can also be key to meeting the energy and channel capacity constraints of a wireless sensor network system. The energy efficiency of the system can be improved by utilising a domain model in the process of converting data into increasingly distilled and high-level representations. Finally, domain models help early detection and reduction the amount of ineffective data forwarding across the network, rather than sustaining the energy expense of transmitting ineffective messages further along the path to the destination.

9.2 Related Work

Wireless sensor network applications that incorporate the special characteristics of the environment in which they operate are starting to appear on the horizon. Hefeeda and Bagheri (2007) present the key aspects in modelling forest fires by analysing the Fire Weather Index (FWI) system to provide efficient fire detection systems. FWI estimates fuel moisture and generates a series of relative fire behaviour indices based on weather observations over decades. This early detection of forest fires system uses the historical statistics given by some components of the FWI and compares it to the weather conditions measured by sensors deployed in the forest to determine the probability of fire ignition. However, in order for the information given by the FWI system to be useful for the forest fire application, certain accuracy requirements on measuring weather conditions are essential as the accuracy and distribution of the sensors impact the accuracy of the FWI system. Another major weakness of this work is that it is only capable of using the information given by the FWI and requires major program modifications if any other sources of environmental knowledge are to be used by the system. Furthermore, it only utilises static environmental information

to decide on application data without using this available information to improve the quality of the collected data as well as the performance and effectiveness of the network system.

Similar to the approach presented in (Hefeeda and Bagheri, 2007), the authors of the BBQ approach (Deshpande et al., 2004) focus on using probabilistic models of the real-world to provide approximate answers efficiently. Probabilistic models are utilised as a framework for optimising the acquisition of sensor readings such that sensors are used to collect data only when the model itself is not sufficiently rich to answer a query with required confidence. In contrast to our work, efforts such as TinyDB (Madden et al., 2005) and BBQ (Deshpande et al., 2004) have adopted an approach that assumes that intelligence is placed at the edge of the network, such as a sink, which is assumed to be less resource constrained than the sensor nodes. The sink uses a spatio-temporal statistical model of the data to decide when to interrogate new readings from sensor nodes; data is refreshed from remote sensors whenever the certainty intervals on the model estimates exceed the uncertainty threshold pre-defined in the query. An interrogation-based approach can not guarantee that all anomalies will always be detected, since the anomaly may occur between two consecutive interrogations. Further, increasing the interrogation frequency to increase anomaly detection probability will increase the data acquisition costs in the network. Finally, this approach was found to be effective with stable network topologies. In highly dynamic network topologies the cost of checking whether the estimation is accurate becomes excessively high. Such a model will require collecting values of all attributes at one location at each time step, and the cost of doing so will most likely reduce any savings in source-sink communication that might result.

Motivated by BBQ, Ken (Chu et al., 2006) exploits the fact that physical environments frequently exhibit predictable stable and strong attribute correlations to improve compression of the data communicated to the sink

node. The basic idea is to use replicated dynamic probabilistic models to reflect the state of the environment being monitored. This is done by maintaining a pair of dynamic probabilistic models over the sensor network attributes with one copy distributed in the sensor network and the other at a PC sink. The sink computes the expected values of the sensor network attributes according to the defined prediction model and uses it to answer queries. When the sensor nodes detect anomalous data that was not predicted by the model within the required certainty level, they route the data back to the sink. This approach is subject to failure as basic suppression. It does not have any mechanism to distinguish between node failure and the case that the data is always within the error bound. Ken is not robust to message loss; it relies on the Markovian nature of the prediction models to presume that any failures will eventually be corrected with model updates, and the approximation certainty will not be affected by the missed updates. They propose periodic updates to ensure models can not be incorrect indefinitely. This approach is not suitable for raw value reconstruction; for any time-step where the model has suffered from failures and is incorrect, the corresponding raw value samples will be wrong. Finally, as the approach presented in (Deshpande et al., 2004), Ken can only handle static network models and does not make use of redundancy.

To the best of the author's knowledge there is no model-driven sensor network framework that exploits the information known about the physical environment for improving the sensor network system efficiency (e.g. reduce energy consumption) and robustness (e.g. detection of node failures). All of the approaches reviewed here (Madden et al., 2005; Hefeeda and Bagheri, 2007; Deshpande et al., 2004; Chu et al., 2006) only employ statistical models that require data collection and processing for building the prediction models mainly to reduce data communication. Furthermore, some of these models are one-dimensional and do not make use of the correlations between different dimensions (sensed attributes). We propose the exploitation of any

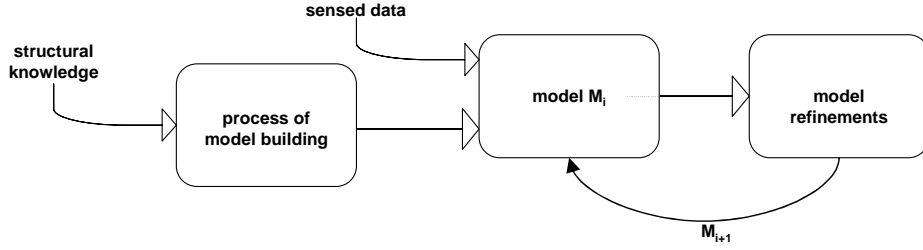


Figure 9.1: Merging of inductive and deductive methods

available information about the physical environment, data correlations, or any other information defined under the domain model. Our framework starts with an initial model then adapts and updates itself to produce a more precise image of the real world through a training procedure.

9.3 M-DAD Mapping Framework Details

In wireless sensor network applications, the involved application domains have a significant effect on the applications performance. The lack of information given by the domain model in designing and implementing wireless sensor network systems may reduce the value of the results produced by such systems and make it harder to extract hidden information from the returned data. For instance, data is dynamically refreshed from remote sensors in response to mutated environmental conditions such as opening a door in a ship chamber. In this section we discuss how incorporating the special characteristics of an application domain helps in developing a mapping service that generates maps which are an accurate reflection of the monitored phenomena status.

The proposed mapping framework, M-DAD, utilises a blend of both inductive and deductive models to establish a successful mapping between sensed data and universal physical principles. This blend is based on the fact that wireless sensor network applications are characterised by the tasks

they are required to perform and the domains they operate within. For example, in wireless sensor networks applications designed to track forest fires (Hefeeda and Bagheri, 2007; Hartung et al., 2006), factors such as the wind speed and direction, topography, and the type of forest have a big influence on the application. Some of these factors can not be sensed or the deployed hardware platform does not have suitable sensing equipment to gather this information. However, this information is very easy to acquire from external resources such as the Geographic Information System (GIS), geology databases, and weather stations and can be integrated into the application to assure the consistency of the theory with the experimental data collected by the network. Therefore, it is desirable to combine the advantages of both methods.

Deductive methods rely on a precise application environment and the explicit knowledge, called structural knowledge, of the underlying domain using first principles to create a model of the problem typically yielding governing equations (Hertkorn and Rudolph, 1999). On the other hand, inductive methods utilise experimental data as the only source of available knowledge (Hertkorn and Rudolph, 1999). Some applications can only be treated using experimental data knowledge due to the lack of other application domain knowledge. Nevertheless, the use of inductive information helps in the generation of data consistency checks based on the structural knowledge abstractions given in the domain model. However, applications have been observed to perform significantly better if they use a combination of the two methods (Hertkorn and Rudolph, 1998). Figure 9.1 shows how the inductive and deductive methods can be merged to capture the advantages of both methods. After the structural knowledge is fed into the system model (deductive process), the sensed data is used to refine and complement the basic structural model of the application domain. This refinement procedure can be done continuously throughout the network life to keep consistent mapping between the physical model and the sensed data.

M-DAD makes use of other knowledge encapsulated in the domain model in the map generation process. Knowledge from domain models provides guidance for map generation from a multi-dimensional data set and has the potential to significantly speed up the map generation process and deliver more accurate maps. To the best of our knowledge, only few previous studies have considered the domain model. For instance, Hofierka et al. (2002) incorporate the topographic characteristics to give better daily and annual mean precipitation predictions. However, there is no unified framework that incorporates all of the information provided by the domain model. In many wireless sensor network applications, conventional data mapping methods are typically not sufficient for mapping sophisticated data patterns from complex, large-scale sensed data and knowledge provided by the domain model with a variety of representations and granularity levels. The general lack of an appropriate data mapping framework for exploiting these rich sensed data resources motivates this work.

Besides exploiting the domain model in map generation, the M-DAD mapping framework performs mapping from related multiple types of the sensed data to overcome the limitations of generating a map from a single sense modality. A single sense modality map typically reveals only a small number of aspects of the monitored phenomena and is unable to infer the correct relations among other types in multi-modal sensing applications. In addition to this inherent data deficiency, high-throughput data sets also often contain errors and noise arising from imperfections of the sensing devices which further obstructs mapping effectiveness. Maps generated from a combination of different types of data are likely to lead to a more coherent map by consolidating information on various aspects of the monitored phenomena. Additionally, the effects of data noise on generated maps will be dramatically reduced, assuming that sensing errors across different data sets are largely independent and the probability that an error is supported by more than one type of data is small. A natural approach to making

use of the relation between the multiple types of sensed data to generate a map is to combine the maps generated from different types of data. We may combine the maps in different methods such as accepting a value at an observation point only when it is commensurate with all maps as defined in the given model. More interestingly, and perhaps with guarantees of delivering better maps, multiple types of data can be analysed concurrently under an integrated relational model. The latter method is novel in the sense that most existing n -dimensional interpolation schemes are defined by applying one-dimensional interpolation in each separate coordinate dimension without taking advantage of the known relations between diverse dimensions (Johnson, 2006).

Multivariate Spatial Interpolation in M-DAD

Most spatial data interpolation methods are based on the distance between the interpolation location P , where the interpolation function has to be determined, and the given set of data points. The M-DAD defines a new metric for distance, suitable for higher dimensions, in which the concept of closeness is described in terms of relationships between sets rather than in terms of the Euclidean distance between points. Using this distance metric, a new generalised interpolation function f , that is suitable for an arbitrary number of variables, is defined.

In multivariate interpolation every set S_i corresponds to an input variable i.e. a sense modality, called i , and referred to as a dimension. In M-DAD, the distance functions do not need to satisfy the formal mathematical requirements for the Euclidean distance definition. The power of such a generalisation can be seen when we include the time variable as one dimension. The spatial data interpolation problem can be stated as follows: Given a set of randomly distributed data points

$$x_i \in \Omega, i \in [1, N], \Omega \subset \mathbb{R}^n \quad (9.1)$$

with function values $y_i \in \mathbb{R}$, and $i \in [1, N]$ we require a continuous function $f : \Omega \longrightarrow \mathbb{R}$ to interpolate unknown intermediate points such that

$$f(x_i) = y_i \text{ where } i \in [1, N] \quad (9.2)$$

We refer to x_i as the observation points. The integer n is the number of dimensions and Ω is a suitable domain containing the observation points. When rewriting this definition in terms of relationships between sets we get the following:

Lemma 9.3.1.1 Given N ordered pairs of *separated* sets $S_i \subset \Omega$ with continuous functions

$$f_i : S_i \longrightarrow \mathbb{R}, i \in [1, n] \quad (9.3)$$

we require a multivariate continuous function $f : \Omega \longrightarrow \mathbb{R}$, defined in the domain $\Omega = S_1 \cup S_2 \cup \dots \cup S_{n-1} \cup S_n$ of the n -dimensional Euclidean space where

$$f(x_i) = f_i(x_i) \forall x_i \in S_i \text{ where } i \in [1, n] \quad (9.4)$$

Proof of Lemma 9.3.1.1 The existence of the global continuous function f can be verified as follows. First, the data set is defined as

$$S = \left\{ \begin{array}{cccc} v_1^{(0)}, & v_1^{(1)}, & \dots, & v_1^{(n')} \\ v_2^{(0)}, & v_2^{(1)}, & \dots, & v_2^{(n')} \\ \vdots & \vdots & & \vdots \\ v_n^{(0)}, & v_n^{(1)}, & \dots, & v_n^{(n')} \end{array} \right\} \quad (9.5)$$

where $n' \leq N$ and $v_i = (x_i, r_i)$, $i \in [1, N]$ and r_i is a reading value of some distinctive modality (e.g. temperature). Let Φ be a topological space on S and there exists open subsets $S_i, i \in [1, n]$

$$\begin{aligned} S_1 &= \{v_1^{(0)}, v_1^{(1)}, \dots, v_1^{(n')}\} \\ S_2 &= \{v_2^{(0)}, v_2^{(1)}, \dots, v_2^{(n')}\} \\ &\vdots \\ S_n &= \{v_n^{(0)}, v_n^{(1)}, \dots, v_n^{(n')}\} \end{aligned} \quad (9.6)$$

which are topological subspaces of Φ such that

$$\Phi_{S_i} = \{S_i \cap U | U \in \Phi\} \quad (9.7)$$

Also define Ψ as a topological space on the co-domain \mathbb{R} of function f . Then there exists a function, f , that has the following properties:

1. Let $f : S_1 \cup S_2 \cup \dots \cup S_{n-1} \cup S_n$ be a mapping defined on the union of subsets $S_i, i \in [1, N]$ such that the restriction mappings $f|_{S_i}$ are continuous. If subsets S_i are *open* subspaces of S or *weakly* separated, then there exist a function f that is continuous over S (proved by Karno (1992)).
2. If $f : \Omega \rightarrow \Psi$ is continuous, then the restriction to $S_i, i \in [1, N]$ is continuous (property, see (Bourbaki, 1966)). The restriction of a continuous global mapping function to a smaller local set, S_i , is still continuous. The local set follows since open sets in the subspace topology are formed from open sets in the topology of the whole space.

Using the point to set distance generalisation, the function f can be determined as a natural generalisation of methods developed for approximating uni-variate functions. Well-known uni-variate interpolation formulas are extended to the multivariate case by using *Geometric Algebra* (GA) in a special way while using a point to set distance metric. Burley et al. (2006) discuss the usefulness of GA for adapting uni-variate numerical methods to multivariate data using no additional mathematical derivation. Their work was motivated by the fact that it is possible to define GAs over an arbitrary number of geometric dimensions and that it is therefore theoretically possible to work with any number of dimensions. This is done simply by replacing the algebra of the real numbers by that of the GA. We apply the ideas in (Burley et al., 2006) to find a multivariate analogue of uni-variate interpolation functions. To show how this approach works, an example of Shepard interpolation of this form is given below:

Given a set of n distinct points $X = \{x_0, x_1, \dots, x_n\} \subset \mathbb{R}^s$, the classical Shepard's interpolation function is defined by

$$(S_{n,\mu}^o f)(x) = \sum_{k=0}^n w_k(x) f(x_k) \quad (9.8)$$

and

$$w_k(x) = \frac{|x - x_k|^{-\mu}}{\sum_{k=0}^n |x - x_k|^{-\mu}} \quad (9.9)$$

where $|\cdot|$ denotes the Euclidean norm in \mathbb{R}^s .

In the uni-variate case ($s = 1$) and $S_{n,2}^0 f$. The basic properties of $S_{n,\mu}^0 f$ are:

1. $(S_{n,\mu}^0 f)(x_i) = f(x_i), i = 0, \dots, n;$
2. $\text{doe}(S_{n,\mu}^0 f) = 0$, where doe is an abbreviation of degree of exactness.

Scale-based Local Distance Metric

In this section we modify the distance metric defined in Section 9.3 to include the knowledge given by the domain model. The domain model helps to significantly reduce the size of the support nodes set to lower than that of the conventional Euclidean-distance-based interpolation methods. The difference in the size of support nodes set can be several orders of magnitude with increasing problem dimension. The increase in the size of the support set can lead to an increase in the computation and processing times of the interpolation algorithm and leads to the same drawbacks of global interpolation methods discussed in Section 7.2. Therefore, the proposed metric attempts to balance the size of the support set with the interpolation algorithm complexity as well as interpolation accuracy.

We define the term *scale* for determining the weight of every given dimension with respect to an interpolation location P based on a combined

Euclidean distance criteria as well as information already known about the application domain a priori to network deployment. While the term *weight* is reserved for the relevance of a data site by calculating the Euclidean distance between an interpolation location and a particular data point. A special case is when f_i is identical for all S_i which means that all sets have the same scale.

For the purposes of M-DAD we define a new scale-based weighting metric, m_P , which uses the information given by the domain model to alter the distance weighting function to improve the interpolation results when applied to an arbitrary number of dimensions. All data sets are ordered pairs (x_i, y_i) such that x_i is the sampling location and y_i is a measured scalar value associated with x_i .

Given a set of data points S_i , standard spatial data interpolation methods define the interpolation function based on the distance from the interpolation location, P , to C_i where $C_i \subseteq S_i$. In local interpolators, C_i is a small collection of the total set of data points. When C_i is extended to include a large proportion of the data points or the total set of points the interpolation function becomes global. In M-DAD, the set C_i for each dimension contains the nearest points for P using m_P . Symbolically, C_i is calculated as

$$C_i = L(d(P, E_j), \delta(S_i)) \quad \forall E_j \in S_i \quad (9.10)$$

where $i \in [1, n]$, L is a local model that selects the support set for calculating P , d is an Euclidean distance function, E_j is an observation point in the dimension S_i , and $\delta(S_i)$ a set of parameters for dimension S_i . Each dimension can have a different set of parameters. These parameters are usually a set of relationships between different dimensions or other application domain characteristics such as obstacles. When predicting the value of a point in dimension S_i we refer to that dimension as S_P .

In uni-dimensional distance weighting methods, the weight, ω can be

calculated as follows

$$\omega = d(P, E_j), E_j \in S_i \quad (9.11)$$

This function can be extended to multi-dimensional distance weighting systems as follows

$$\omega = K(P, S_i), \quad i \in [0, n] \quad (9.12)$$

where $K(P, S_i)$ is the distance from the interpolation position P to data set S_i and n is the number of dimensions in the system. Equation 9.12 can now be extended to include the domain model parameters of arbitrary dimensional system. Then the dimension-based scaling metric can be defined as

$$m_P = \sum_i L(K(P, S_i), \delta(S_i)) \quad i \in [0, n] \text{ and } S_i \neq C_P \quad (9.13)$$

where C_P is the dimension containing P .

9.4 Distributed Self-Adaptation in the Mapping Service

Benefits of Self-Adaptation

Wireless sensor network deployments are characterised by frequent network system changes. These changes can be internal, e.g. changes in nodes energy level, or external, e.g. topographical changes in the application domain. Internal and external changes have an imperative influence on the network performance, maintenance, and the accuracy of the returned results. Since wireless sensor networks on-site maintenance and reconfiguration is not always feasible, self-adaptation to external as well as internal system changes is crucial for enabling a successful deployment of a wireless sensor network mapping service.

Adaptation to internal changes is a well-established research area and the reader is referred to (Virrankoski, 2005; Cerpa and Estrin, 2004; Minder et al., 2007; Handy et al., 2002) and references therein. However, adaptation to external physical environmental changes in the application domain still needs in-depth investigation. The M-DAD mapping framework can meet the goal of reliability and reactivity, and demonstrates satisfactory robustness and relatively longer network lifetime using the information collected about the external environment they operate within. Sensed data is not only useful for end users, but can be valuable for network services as well. For example, during emergency situations such as a ships fire, evacuees can have difficulty finding safe escape routes out of the ship. These difficulties can be a result of the fire filling the escape route with smoke or parts of the ship collapsing, blocking the path to safe egress routes. In such conditions, it is difficult to find a safe way out of a ship, and a map as to which of the escape routes are safe and how to get to them can be very valuable for the evacuees. On the other hand, a change in the environmental conditions, e.g. collapsed walls, may provoke a change in the network functionality. For example, when the temperature goes above 60°C, turn the smoke sensors off or when topographical changes are detected start a new data transmission phase.

In this section we extend the capabilities of M-DAD to overcome challenges imposed by the described external system changes through applying self-adaptation intelligence to continuously adapt to erratic changes in the application domain conditions. Dynamically adaptive mapping includes tasks that detect external system changes and tasks that update the domain model with the new environmental conditions. At run time, M-DAD integrates the sensory data with contextual information to update the initial application domain model provided by network owners to maintain a coherent logical picture of the world over time.

Self-adaptation is particularly useful in long-term wireless sensor net-

work deployments where the environmental conditions change significantly over time, necessitating the update of the domain model to reflect the changes in the contextual knowledge provided by the physical constraints imposed by the local environmental conditions where sensors are located. Self-adapting mapping will provide high reliability and predictability for years at a time without constant tuning of the domain model by wireless experts. This allows mapping services to evolve at run-time with less intervention of the user and leads to near-optimal and flexible design that is simple and inexpensive to deploy and maintain. This adaptation procedure will recover, over time, from the effects of user domain modelling inaccuracies.

We realise that self-adaptation is a challenging problem and considerable work is being done by the research community in that area. However, in this work we aim to deal with a small set of adaptivity issues that have a significant effect on the mapping services.

Adaptability Implementation in M-DAD

To implement adaptability in M-DAD, the interpolation capability of the network is exploited to perform local training of the map generation service. Each node uses the readings of its surrounding nodes to predict its own reading value (y') using the scale-based metric defined in equation 9.13. Then, y' is compared to the node measured value, y . It is desirable that the estimate of y' minimises the Standard Deviation σ

$$\sigma = \sqrt{(y' - y)^2} \quad (9.14)$$

Nodes modify the size of the support set to include the minimum number of nodes needed to predict y with a certain level of accuracy. Furthermore, in multi-dimensional applications, nodes will change the weight of each dimension to improve the prediction accuracy of y' . In fact, nodes will alter the relationships between different dimensions initially given in the

domain model in order to recover the effect of inaccuracies in the given domain model or to adapt to emerging environmental changes. These model updates influence the estimation results because y' is calculated using the scaling metric m_p (eq. 9.13). Finally, a prediction accuracy criterion, Δ , is defined as the average σ_j where

$$\sigma_j = \sum_i \sqrt{(y_i - y'_i)^2} \quad j \in [1, n] \text{ and } i \in [1, N] \quad (9.15)$$

where n is the number of dimensions and N is the number of readings in dimension j . Then Δ is written as

$$\Delta = \frac{\sum \sigma_j}{n} \quad j \in [1, n] \quad (9.16)$$

Likewise the one dimensional case, Δ must always be minimised to achieve the best mapping results.

However, when individual nodes alter their programmed domain model independently from the network, the mapping service may become unstable because of the inconsistency in the domain model defined on various nodes. Such inconsistencies may lead to inconsistent system states and conflicting differences in calculating mapping values. Furthermore, some of the detected environmental changes may result from sensing faults, which are hard to detect without collaboration with neighbouring nodes. To ensure mapping stability and overcome such concerns, we propose a *Virtual Congress Algorithm* to manage global model updates locally. This algorithm is discussed in detail in Subsection 9.4.

The Virtual Congress Algorithm

Sensor nodes make use of sense data to detect changes in the environment. After sensor nodes forward their readings to upper layers of the network topology, each node periodically checks its attached sensors readings against the domain model. If a reading does not comply with the domain model, e.g.

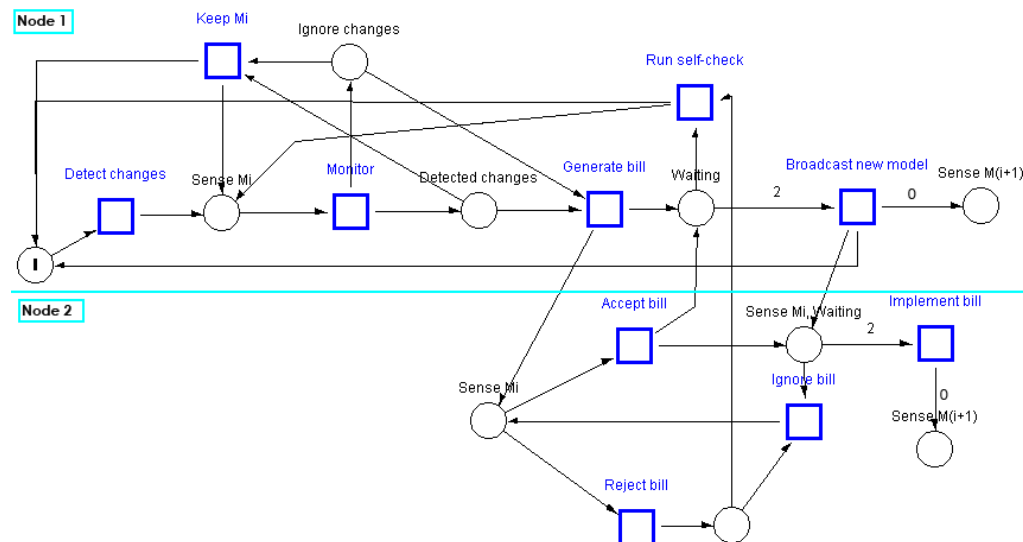


Figure 9.2: A Petri-net that models the virtual congress algorithm.

temperature reading equals 80°C and the relative humidity equals 80%, then the sensor runs a self-check service such as those proposed in (Wan et al., 2008; Babbitt et al., 2008b; Boonma and Suzuki, 2007; Pietro et al., 2008). In this case, the node may run a sensors calibration service. The sensor node uses its neighbourhood readings for calibrating its sensors and to decide whether both sensors need calibration. In other situations, such as when using one sense modality to estimate other sensed modalities, the sensor may find, overtime, that changing the defined relationships may result in a smaller Δ . This node then collaborates with other nodes in the network to test the correctness of the domain model. This process allows the recovery from modelling inaccuracies and discovery of new relationships which is an important scientific goal.

The *Virtual Congress Algorithm* (VCA) top-down technique is proposed to manage the global and local domain model updates. VCA provides a high-level collaboration environment in which the system can achieve globally efficient behaviour under dynamic environmental conditions. Instead

of re-programming individual nodes, the network is viewed as a virtual congress where nodes are *senators* who vote for legislating changes to the domain model in response to locally detected environmental conditions. This algorithm is an attractive solution as senators collaboratively decide upon their local knowledge on the behaviour and correctness of the system. Logically related nodes, *chambers*, are granted some power to impute the local changes, *federal decisions*, that is not detected by all nodes in the network. In cluster-based networks, a cluster can be defined as a chamber. A senator may introduce a proposal in the chamber as a *bill*. To prevent overloading the chamber with proposals, each senator must monitor the changes over time using equation 9.16 before putting them into a bill. Besides, no two or more identical bills are allowed to be proposed by two different senators. When a node receives two identical bills from two different senators it only considers the first one it receives. Senators evaluate and study the proposed bill and send their voting results accordingly to the proposing senator. Upon receiving the required number of votes v , the proposing senator disseminates the bill to the chamber and all nodes implement the new changes that have been agreed on. The value of v was empirically estimated to be over 50% of chamber population because it helps to avoid false positives (see Experiment 9.5 for empirical analysis). If the bill gets rejected, then the proposing senator suppose that there is a local problem and starts a fault detection or recalibration service. Once a bill is approved by one chamber, it is sent to other chamber heads who may accept or reject it. In order for the bill to become a *state law*, all chamber heads must agree to identical version of the bill. When the bill is submitted to the *president*, the sink node, he may choose to sign the bill, thereby making it *law*. The president informs all chamber heads whether the bill became a law or not. If the chamber head does not receive a response from the president within a specific period of time, then it register the bill it generated as rejected.

Figure 9.2 shows a Petri net that models the behaviour of the VCA in

a chamber that consists of two sensing nodes (senators). In this model the first senator, Node 1, detects a change in the environmental conditions. The place with a single token is a guarding condition that allows the senator to generate only one bill at any time. When a senator detects a change, it enters into a monitoring phase after which it decides to keep the existing local domain model or to generate a bill to update that model. When a senator collects the required number of votes (in this scenario $v = 2$), it broadcasts the new model which is then implemented by all nodes in the chamber. Finally, a token is deposited into the start place after the completion of the implement model update activity.

This Petri net model was simulated using the S/T Petri-Net simulation system (Braunl, 2005). On simulation, the useful information like liveness and deadlocks if any, are determined. It was found that the VCA Petri net is live and deadlock-free. The simulation of the VCA Petri net with non-deterministic selection of activities allows checking the algorithm against deadlocks caused by controllers, introduced to enforce the given specifications, especially in the closed-loop net branches.

9.5 Experimental Evaluation

While no single domain of scientific endeavour can serve as a basis for designing a general framework, an appropriate choice of specific application domain is important in providing significant insights relating to requirements of such a mapping framework. Two important criteria in choosing a working case study application domain are that chosen applications involve phenomena of major importance from a scientific point of view and that they involve a set of complex relations between different variates. Therefore, to illustrate the benefits of exploiting the domain model in map generation we consider heat diffusion in metals model.

Heat diffusion describes the distribution of heat in a given area over

time. We consider a rigid, thin, heat-conducting bar which means that the temperature of the bar does not vary with the thickness of the bar. The heat diffusion equation is defined as

$$\frac{\partial \phi(x, t)}{\partial t} = D \nabla^2 \phi(x, t) \quad (9.17)$$

where $\phi(x, t)$ is the temperature at position x at time t , $\frac{\partial \phi(x, t)}{\partial t}$ is the rate of change of temperature at a point over time, D is a constant that represent the diffusion coefficient, and $\nabla = \frac{\partial^2 u}{\partial x^2}$.

The heat equation is of fundamental importance in diverse scientific fields. For example, in mathematics, it is the prototypical parabolic partial differential equation. The heat diffusion equation arises in connection with the study of chemical diffusion and other related processes.

In the following set of experiments the effective thermal diffusivity in a cargo ship is studied. Some aspects of a cargo ship fire were modelled, particularly, heat diffusion in the metal body of the ship. This model contains many interesting environmental and layout features such as hatches or doors. This allows us to test how M-DAD can benefit from such information in the mapping process. Moreover, the environmental changes can be dynamic, thus, the VCA can be evaluated. It also involves many classes of complex, spatio-temporal variables and relatively complex relationships among different dimensions. For instance, the domain model describes the relationship between the temperature and humidity levels. This relationship can be used to test how M-DAD can generate better estimations for one sense modality, e.g. temperature, using other independent sense modality, e.g. humidity. The temperature-humidity relationship is expressed as the amount of water vapour present in the air divided by the amount of water vapour required to saturate air at that temperature. One of the most important classes of queries about such systems relate to level of temperature in various parts of the ship at different times before, during and after an event such as power failure or rainfall storms. Heat diffusion occurs slowly

which allows the study of the M-DAD without the constraints of timeliness. Finally, heat diffusion has many other real life applications such as airplane body conditions monitoring, oil pipes monitoring, etc.

The model was restricted to a chamber in the ship which has two large doors that can be opened by the ship crane. The chosen chamber was modelled by a brass sheet which contains a hole segment excavation to model an opened door. A simple domain model was defined to carry information about doors that when opened they impact the heat diffusion in the ship. The hole segment that represents an opened door was excavated in the brass sheet with 0.01cm width, 2cm length, and end points $A(12, 0)$ and $B(12, 3.6)$. The hole segment width defines the door strength. The used brass sheet is a $7.5\text{cm} \times 24\text{cm}$ rectangle and is 1mm thick.

Experiment 1: Incorporation of the Domain Model in the Mapping Services

Aim: The aim of this experiment is to study the effect of integrating the knowledge given by the domain model into the mapping service.

Procedure: A FLIR ThermaCAM P65 Infrared (IR) camera (FLIR Systems, 2008), Figure 9.3, was used to take sharp thermal images and produce an accurate temperature analysis and results. The ThermaCAM P65 camera delivers 320×240 IR resolution (640×480 pixels, full colour) at 0.08°C thermal sensitivity. Also, 10 Toradex Oak USB Sensor (AG, 2008) nodes equipped with humidity (0 to 100 %RH) and temperature (-40°C to 85°C) sensors were distributed over the brass sheet. Finally, a 1371°C blue flame was placed on the middle of one edge of the brass sheet as a heat source. Brass (an alloy of copper and zinc) sheet was chosen because it is a good thermal conductor and allows imaging within the temperature range of the available IR camera with less reflection than other metals such as Aluminium and Steel. The thermal diffusivity of brass is 109W/mK at 300K . Figure 9.4 shows the brass sheet with the excavated hole segment, the sensors, and the



Figure 9.3: FLIR iInfracAM SD infrared camera used for taking thermal images of the brass sheet (FLIR Systems, 2008).

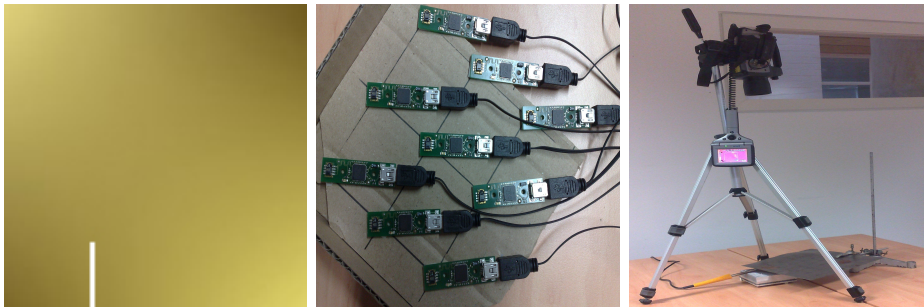


Figure 9.4: A brass sheet with segment hole excavation; Toradex Oak USB sensors; and the experimental apparatus.

IR camera apparatus.

Using the experimental setup described above, the first experiment was performed using the brass sheet before the hole segment excavation. After 30 seconds of applying the heat, thermal measurements from the Toradex Oak sensors were recorded in addition to a thermal image for the whole sheet taken by the ThermaCAM P65 IR camera. The mapping service was

run over a subset of the data collected from this experiment to observe how the heat will diffuse in the brass sheet in the absence of any obstacles. The map produced by the mapping service will be compared to the camera thermal maps to examine the map generation accuracy level.

The same experiment was then repeated on the sheet with the segment hole excavation and sensor thermal measurements as well as a camera thermal image were taken after applying heat on the brass sheet. The mapping service was run using the same size of the thermal data-set used in the previous experiment. Three experimental mapping service runs were performed: (1) The mapping service does not have any domain model knowledge. Particularly, the presence of the obstacle and its characteristics. (2) The mapping service integrates *some* knowledge given by the domain model. Particularly, the presence of the obstacle, its position, and length. (3) The mapping service integrates all the knowledge given by the domain model. Particularly, the presence of the obstacle, its position, length, and strength.

Firstly, the mapping results from run (1) are compared with the results from run (2) to observe if the integration of the domain model knowledge can improve the mapping performance. Secondly, the results from run (2) are compared with the results from run (3) to observe whether the mapping performance is going to improve as more knowledge about the domain model is being utilised by the mapping service.

Results and discussion: Figure 9.5 shows the heat diffusion map generated by the FLIR ThermaCAM P65 IR camera. Given that the heat is applied at the middle of the top edge of the brass sheet and the location of the obstacle, by comparing the left side and right side areas around the heat source, this figure shows that the existence of the obstacle has an effect on heat diffusion through the brass sheet. It was observed that the obstacle strongly reduced the temperature rise in the area on its right side. This map has been randomly down-sampled to 1000 points, that is 1.5% of the total 455×147 to be used by the mapping service to generate the total heat



Figure 9.5: Heat diffusion map taken by ThermaCAM P65 Infrared (IR) camera.

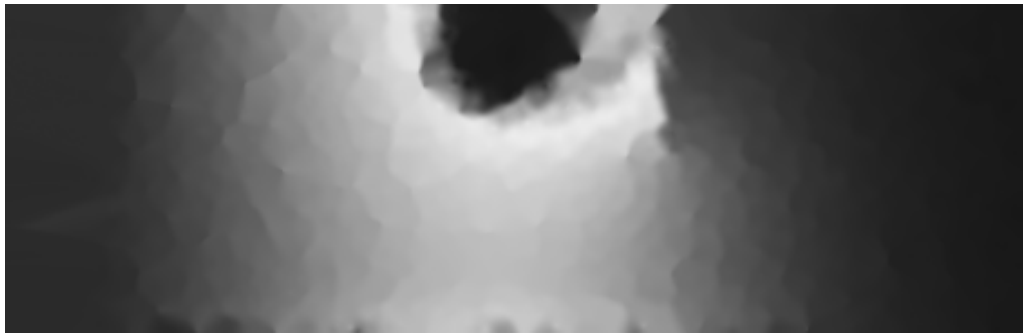


Figure 9.6: Heat map generated by the distributed mapping service defined in Chapter 8.

map.

Figure 9.6 shows the map generated by the distributed mapping service defined in Chapter 8. Compared with Figure 9.5, the obtained map conserves perfectly the global appearance and many of the details of the original map with 98.5% less data. However, the area containing the obstacle has not been correctly reconstructed and has caused hard edges around the location of heat source. This is due to attenuation between adjacent points and the fact that some interpolation areas contain many sensor readings



Figure 9.7: Interpolated heat map generated by M-DAD given obstacle location and length.



Figure 9.8: Interpolated heat map generated by M-DAD given obstacle location, width and length.

with almost the same elevation. That asserts that modifications to the map generation service are sometimes needed in order to interactively correct the mapping parameters.

Figure 9.7 shows the map generated by the M-DAD mapping framework. M-DAD was given some information about the application domain including the existence of the obstacle, its location and length. M-DAD replaces the Euclidean distance function from P to D_i with the distance from P to the nearest obstacle endpoint E plus the distance from E to D_i . It is

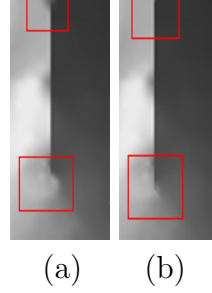


Figure 9.9: (a) Detailed section of Figure 9.7 showing the area around the obstacle with artifacts marked in red. (b) Detailed section of Figure 9.8 showing the area around the obstacle with reduced artifacts.

observed that the map obtained by M-DAD conserves perfectly the global appearance as the distributed mapping service (Figure 9.5). However, using the given local semantics, M-DAD reduced the prediction error and visually it accurately captured the effect of the heat obstacle on heat diffusion through the brass sheet. The M-DAD generated map is smoother than that rendered with the distributed mapping services, especially in some sub-regions containing the obstacle and around the heat source location. Moreover, M-DAD reduced the heat diffusion prediction error in the area on the right of the obstacle which was not captured accurately in Figure 9.6.

Figure 9.8 shows the map generated by M-DAD with a more complex domain model than the previous M-DAD version (Figure 9.7). Particularly, in this version we give M-DAD the obstacle width. A better approximation to the real surface near the obstacle is observed. The new details included in the domain model removed artifacts from both ends of the obstacle as shown in Figure 9.9. This is due to the inclusion of the obstacle width in weighting sensor readings when calculating P which further reduces the effect of geographically nearby sensors that are disconnected from P by an obstacle.

Conclusion: This experiment shows that the incorporation of the domain

model in the mapping service significantly improves the distributed mapping service map production quality. In terms of visual aspects, M-DAD has represented the obstacle accurately and it has generated an approximation function with a smooth shape despite discontinuities in the mapped surface. The mapping quality has been further improved with the increase of information given by the domain model. The results obtained confirm that M-DAD is an effective mapping framework for such a complicated heat transfer fields though further considerations are needed for predicting turbulence just behind the obstacle.

Experiment 2: Mapping from Related Multiple Dimensions

Aim: The aim of this experiment is to study if mapping from related multiple types of the sensed data can lead to an improved mapping performance by overcoming some of the limitations of generating a map from a single sense modality.

Procedure: 10 Toradex Oak USB Sensors equipped with humidity and temperature sensors were placed over the brass sheet. Cold water was sprayed onto the brass sheet to increase the humidity in order to make relationships between the temperature and humidity more visible. Then, a blue flame was placed on the middle of the top edge of the brass sheet. Finally, the following steps are applied:

1. remove one temperature reading from the collected data set
2. use the distributed mapping service to calculate the removed temperature reading using the rest of the data set
3. use M-DAD to calculate the removed temperature reading using the rest of the data set

4. calculate the standard deviation for the temperature value resulted from 2 and 3
5. repeat steps 1 to 4 for each of the 10 sensors temperature readings

Results and discussion: Figure 9.10 plots the humidity and temperature readings collected by the Toradex Oak USB Sensors. This figure shows that the temperature is inversely proportional to the humidity. This relationship is expressed as the amount of water vapour present in the air divided by the amount of water vapour required to saturate air at that temperature. M-DAD utilises this relationship between these two dimensions to improve the temperature prediction accuracy using the humidity map. Figure 9.11 shows a plotting of the standard deviation of the calculated temperature values by M-DAD and the distributed mapping service. With the existence of only one discontinuity, M-DAD reduced the standard deviation by between 0.17% and 12% compared to the standard mapping service. This is done by constructing the support set from nodes which are more related to P , for example, nodes that have close humidity reading to that of P .

Conclusion: This experiment proves that mapping from related multiple dimensions can improve the generated map quality. This observation is confirmed by the data shown in Figure 9.11. The results from this experiment and Experiment 1 confirm the general theory of M-DAD defined in Section 9.3.

Experiment 3: Adaptations to Changes in the Domain Model

Aim: The aim of this experiment is the study the effectiveness of the proposed VCA in modifying the domain model to better fit the current state of the application domain. It also aims to study the effectiveness of VCA in differentiating between faulty sensor behaviour and actual changes in the domain model.

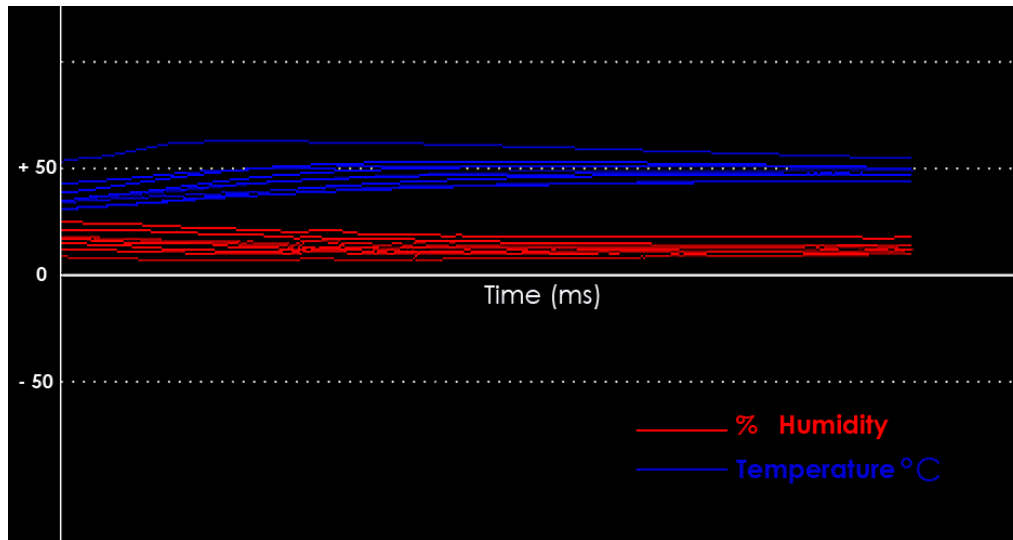


Figure 9.10: Humidity and temperature measurements generated by 10 Toradex Oak USB Sensors.

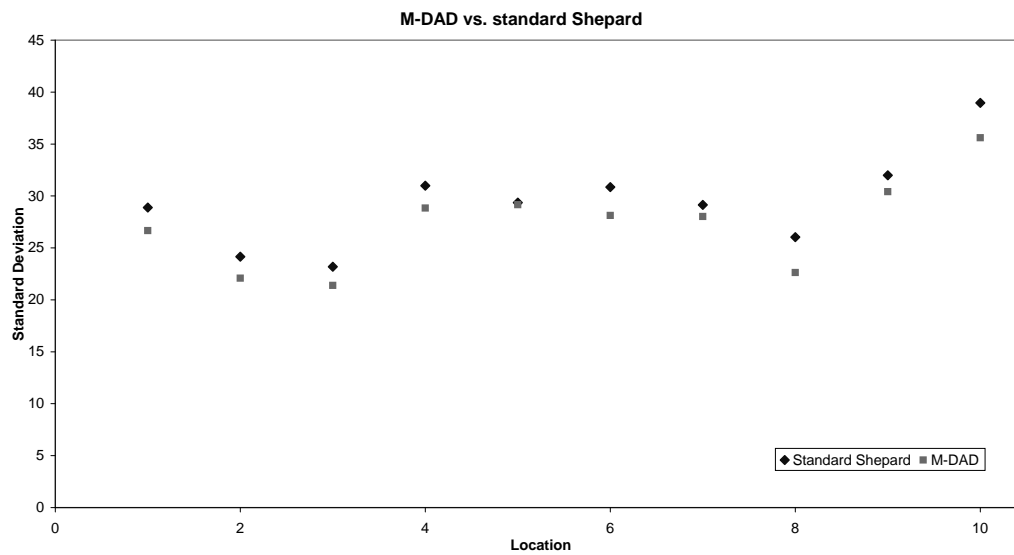


Figure 9.11: The Standard Deviation of temperature values at 10 locations calculated by M-DAD using the humidity map.

Procedure: The same experimental setup described in Experiment 1 was used here. After applying heat on the brass sheet for 30sec, the obstacle length was increased from 2cm to 3.6cm. Then, it was observed how the M-DAD mapping framework will use the VCA to adapt its behaviour to that emerging change in the obstacle length. Particularly, the bill which contains the best (closest to the actual obstacle length) detected obstacle length value, the agreed bills, and the federal laws were examined. The existence of an obstacle can be detected by machine vision techniques or IR motion detection sensors which are not available on the Toradex Oak sensors used here. Wireless communications breaks caused by an obstacle attenuation are hard to predict, but can be estimated using published metrics. For example, Airespace design notes (Extricom, 2007) estimate 2.4GHz (802.11b/g) loss for drywall at 4dB, brick wall at 8dB, and concrete wall at 10 – 15dB. When a signal collides with an obstacle, the level of attenuation depends strongly on which type of material the obstacle is made from, for example, metal obstacles tend to reflect a signal, while water absorbs it. Therefore, metals have a very high degree of attenuation (Kioskea.net, 2008). It was assumed here that the obstacle is continuous and the existence of this obstacle between two directly communicating nodes will break the wireless links between them. The local semantics of the application domain were defined to interpret the break of direct wireless links between two nodes while being able to communicate through an intermediate node(s) as *there exists an obstacle between the two communicating nodes*.

Secondly, 30% of the nodes were configured to act as faulty nodes in each experimental run and v was assigned different values. Faulty nodes were allowed to propose bills that carry incorrect changes in the application domain. Finally, it was studied how the choice of v will affect the VCA ability to discover bills generated by faulty nodes.

Results and discussion: Table 9.1 shows three M-DAD mapping frame-

work runs each with different node distributions. Three different randomly distributed node topologies were tested because the accuracy of obstacle detection according to the described model is highly dependent on the nodes location and density around the obstacle. The number of cluster heads in each run was 48, 49 and 49 respectively. Table 9.1 also shows the number of proposed bills, the best proposed bill and the agreed bill for each mapping run. The results in Table 9.1 shows that VCA has always detected the obstacle length accurately and that the best proposed bill was not always agreed within the cluster. This is partially due to the cluster formation process which is able to deal with obstacles (see Chapter 6). In MuMHR clustering, nodes joins the nearest cluster head which may result in forming a cluster from nodes around the top of the obstacle where the most accurate obstacle length is detected, however, the cluster may not contain enough nodes that sense the existence of the obstacle (see the example in Figure 9.12). Nonetheless, the average VCA agreed bills in the three mapping runs were 53.91 pixels which is close to the actual obstacle length (60 pixels). Adapting the mapping service to the new obstacle length improves the produced map quality. Figure 9.13 shows the heat map generated by M-DAD using the initial obstacle length. Visually, this map does not totally reflect the current state of the application domain compared to Figure 9.7 (map with obstacle length equal to 60 pixels). Quantitatively, the RMSD difference from Figure 9.7 was increased by 1.0.

In the three mapping runs, zero bills became state laws. This is because all the changes in the application domain were local to part of the network and the majority of the clusters did not sense these changes. This illustrates the mutual benefit of localising the VCA and distributing it over two levels: the local/cluster level; and the global/network level.

The defined domain model is also highly dependent on the network density because the increase in the number of nodes increases the probability that a change in the domain model is detected. Table 9.2 and Table 9.3

Table 9.1: The obstacle length (in pixels) in the best proposed bill and the agreed bill in three M-DAD mapping runs at 1000 nodes density.

	Num. of proposed bills	Best bill	Agreed Bill
Run 1	17	59.57	52.0
Run 2	19	60.0	57.74
Run 3	14	60.0	52.0

show the results of VCA testing with 100 and 500 network densities each with results from three different random node distributed topologies. Also notice that the simple domain model defined in the previous paragraph can be extended to the discontinuous straight obstacles case by defining discontinuities in the obstacle as: (1) the intersection location between the straight line containing the obstacle with the segment connecting the two directly communicating nodes n_1 and n_2 ; (2) the intersection location between any segment connecting two successive nodes in the communication path tree between nodes n_1 and n_2 .

Table 9.4 shows the number of bills proposed by faulty nodes and how many of those bills were agreed. First, the number of bills generated by the faulty sensors in each experimental run with different values for v were counted. The values tested were $v = 40\%$, $v = 50\%$, and $v = 60\%$. It was found that the $v = 50\%$ gave the minimum number of agreed faulty bills by avoiding the false positives we get with the $v = 40\%$. Whereas the $v = 60\%$ caused many correct bills to be rejected. The results shown in Table 9.4 were generated using $v = 50\%$.

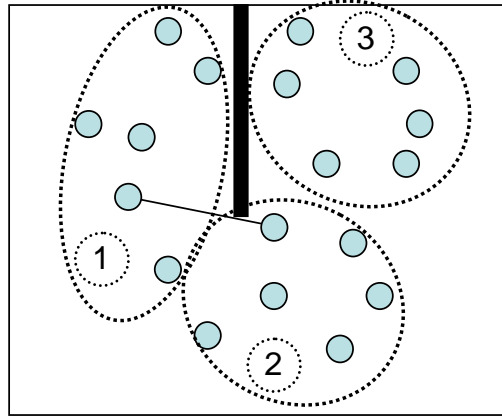


Figure 9.12: A network topology that consists of 3 clusters deployed in an area containing an obstacle. In cluster 2 there is only one node which can detect the obstacle. This node detects the obstacle exactly at its end point and generates the best bill in the network. That bill is not going to be agreed in the cluster because the majority of that clusters members are not aware of the obstacle.

Table 9.2: The obstacle length (in pixels) in the best proposed bill and the agreed bill in three M-DAD mapping runs at 100 nodes density.

	Num. of proposed bills	Best proposed bill	Agreed bill
Run 1	16	43.46	43.46
Run 2	14	53.26	53.26
Run 3	13	48.64	34.07

Table 9.3: The obstacle length (in pixels) in the best proposed bill and the agreed bill in three M-DAD mapping runs at 500 nodes density.

	Num. of proposed bills	Best proposed bill	Agreed bill
Run 1	15	59.33	53.96
Run 2	16	59.82	58.65
Run 3	11	60.0	53.42



Figure 9.13: Heat map generated by M-DAD with 2cm obstacle length.

Table 9.4: The number of bills proposed by the faulty nodes.

	Num of faulty bills	Num of agreed faulty bills
Run 1	2	0
Run 2	2	1
Run 3	1	0

Conclusion: This experiment shows that VCA helps to adapt to some changes in the domain model in a distributed manner. It also shows that the optimal value for v is 50%. This experiment is an instance of the general case studied in Experiment 2, particularly, the nearest neighbour triangulation RF connectivity map is used as one dimension to predict the heat map. Therefore, the improved mapping performance proved in this experiment confirms the results found in Experiment 2.

9.6 Chapter Summary

This chapter proposed M-DAD, a mapping framework, that utilises the knowledge given by the domain model for producing highly accurate maps. M-DAD is capable of dealing with an arbitrary number of dimensions, performs distributed self-adaptation, exploits the application domain model, and generates maps by interpolating from different related sensed data

modalities. The incorporation of the domain model improves the mapping quality in terms of maps predictive error and smoothness.

Adaptive M-DAD spontaneously responds to system changes. In M-DAD, nodes are loaded with an initial model and they collaborate to produce a refined model which reflects the current state of the domain model. This chapter has also demonstrated how the VCA algorithm is applied and experimental results proved that the refined initial model indeed lead to improved mapping performance.

Chapter 10

Conclusion

This thesis has taken an *evolutionary* approach to address the issues of efficient sense data extraction and visualisation using existing data and network potential of a standard sensor network. Rather than focusing only on a single component of the wireless sensor network architecture to improve efficiency, a cross-component approach that maximises the utilisation of information offered by various wireless sensor network components has been presented. It has been illustrated how information generated by a particular component of the sensor network architecture can be used by other components to help improve their operation as well. To illustrate how the proposed cross-component approach can be used, solutions where routing, map generation and self-adaptation operations might benefit by using cross-component information were presented.

Information extraction and visualisation is vital to all practical sensor network applications. It is based on both functionality that displays data and human capabilities that recognise patterns, trends, and relationships. The ‘map’ style of presentation has been identified and proved as a suitable data extraction and visualisation format. This format builds on visual and analytical processes developed in various disciplines with extensions that handle very large multivariate data sets.

As a first step in the research, the feasibility of sense data extraction and visualisation mapping services has been examined. The experiments demonstrate that a mapping service for wireless sensor networks is feasible. The mapping service has been found to be suitable for informing other network services as well as the delivery of field information visualisation. A number of disadvantages to this approach have been identified. One is the high energy consumption coupled with transmitting large amounts of data from nodes in the network to the centralised location. The centralised approach is typically only feasible for small immobile sensor applications, as it fails to adequately address the issues of scalability or of survivability under real world conditions. The damage of the central node or an associated crucial communication link results in total network failure. A further problem is when only a subset of the data is required for information extract. While this would suggest less transmissions which in turn means less energy consumption, the centralised approach imposes that all data be transmitted to the central point for analysis. A large portion of the returned data is not useful, particularly the unchanged modalities and the data from regions out of interest. Therefore, a scalable sense data mapping solution is essential.

The next step was to identify and develop two building blocks for the mapping service: routing and map generation.

It has been found that an efficient routing protocol is vital to the development of a scalable mapping service that requires timely and fault tolerant data delivery. This work has presented an improvement on the implementation of information routing capabilities in ad hoc wireless sensor networks called MuMHR. MuMHR provides multi-hop and multi-path communication which makes it suitable for large scale wireless sensor networks. Furthermore, MuMHR achieves load balancing at both the local/cluster level and at the network level. Improving the protocols used by each sensor node can increase the network's localisation and power conservation abilities. Simulation results demonstrated that MuMHR meets the energy consump-

tion, timeliness, and reliability requirements that have a significant effect on the mapping and other system components performance. Because using a more efficient algorithm will improve the overall effectiveness of the entire network system, the routing network component has been chosen as a working example to illustrate how the mapping service can benefit from information provided by existing network components.

The map generation component enables the development of wireless sensor network applications and dealing with the data in more abstract forms than simply a cloud of points. Sense data are discrete measurements over a finite region of space. It has been proposed that a discrete data type is not the best way to model a surface at an abstract level: the finiteness of the sample should be restricted to the internal representation, while the abstract data model should be continuous. The interpolation process offers the ability to predict an intermediate value of one variable which is a function of a second variable when values of the second variable related to numerous discrete values of the first variable are known. By using the interpolation capability to interpolate inter-node values upon the network, it became possible to build applications that are unaware of the physical reality of sparse data. Empirical analysis has shown that spatial interpolation methods are an accurate, flexible and efficient method for multivariate interpolation of scattered data. Particularly, Shepard interpolation has been shown to be suitable for wireless sensor network applications. Shepard interpolation is an efficient method for multivariate interpolation of large scattered data sets. It requires small storage and it is easy to generalise to additional dimensions. One important advantage of Shepard interpolation is that it can be localised and thus it can be easily integrated in distributed software.

Using the findings at this stage and to respond to the requirements of efficient sense data mapping, a distributed mapping service has been defined. It has been shown how this service can exploit the routing and the map generation capabilities of the network to make mapping applications

simple to develop. A number of experiments incorporating real world data mapping have been conducted and the efficiency of the distributed approach has been analysed against results obtained using the centralised mapping service. With reduction of energy consumption as a key objective of the distributed service, distribution has helped to satisfy a large number of real world requirements such as: (1) In scenarios where it is desirable or necessary to process information on site, a distributed mapping service provides a critical solution to in-field data analysis; (2) It allows the user to obtain a map from sub-regions of the network when needed. The distributed mapping service solves global network data mapping problems through localised and distributed computation. Cluster heads have been selected as the caches for local maps which are cached and merged at the sink node. The global map is updated periodically to mirror the current state of the environment. To reduce the number of update messages, the map generation service has been used to perform training processes to decide whether a change in the environment is significant and must be communicated to the sink. The global map provides a high level interface and abstracts away some of the internal workings of the network. It can also be used to target queries for generating detailed maps from some regions in the network. The experimental results have confirmed that the distributed mapping service produces significant energy savings over the centralised approach.

The distributed mapping service has been elevated on the inherent redundancies and relationships among the gathered sense data, as information about a particular event of interest in a sensor network is usually captured in multiple sensed modalities. This data correlation can be revealed temporally, spatially, and across different sense modalities. The new mapping framework, called M-DAD, utilises the defined correlations to map a distinctive sense modality using different multiple, independent, sense modalities, which results in higher accuracy maps than mapping from a single modality. M-DAD is also capable of mapping unrelated multivariate data.

In many wireless sensor networks applications, a lot of information about the application domain is known in a priori to the network deployment, and thus it can be utilised in the system design and deployment. The utilisation of the domain model has been found to leverage computational power to simulate, visualise, manipulate, predict and gain intuition about the phenomenon being studied. By using the data context encapsulated in the domain model to throw light on its meaning, the mapping service was able to give more meaning to the collected data. The domain model has been used to dynamically minimise the mapping predictive error. To enable mapping in dynamic environments that impose varying functional and performance requirements, a self-adaptation extension has been added to automatically adapt the mapping service behaviour to a large set of environmental changes, to allow accurate mapping with minimum system maintenance. To differentiate between environmental changes and faulty node behaviour, a new distributed algorithm, Virtual Congress Algorithm, has been proposed. This has made M-DAD more resilient to faults.

The mapping framework proposed in this thesis gives strong basis for a number of interesting directions for future work, which will lead to improved levels of mapping accuracy. This work could be extended to other, related areas including:

1. The clear benefits demonstrated in this thesis of exploiting the domain model in map building encourages the author to extend this feature to include more information from other sources, such as CubeSats (Nugent et al., 2008), in real time.
2. Improve the system adaptation capability to incorporate a larger set of internal and external system changes that might have influence on the mapping performance.
3. Extend M-DAD to support the integration of other network components, such as the data aggregation component, and investigate how

the mapping service can benefit from these components and what information given by such components can be beneficial to the mapping service.

Publications from the Thesis

Mohammad Hammoudeh, Robert Newman, and Christopher Dennett, Sarah Mount, A Combined Inductive and Deductive Sense Data Extraction and Visualisation Service. In The 2009 International Conference on Pervasive Services (ICPS'2009), 2009. ACM.

Mohammad Hammoudeh, Robert Newman, and Christopher Dennett, Sarah Mount, Inductive as a Support of Deductive Data Visualisation in Wireless Sensor Networks (Short paper). In IEEE Symposium on Computers and Communications (ISCC'09), 2009.

Mohammad Hammoudeh, Robert Newman, and Christopher Dennett, Sarah Mount, Inductive as a Support of Deductive Data Visualisation in Wireless Sensor Networks (Extended paper). In SENSORCOMM '09: Proceedings of the 2009 Third International Conference on Sensor Technologies and Applications, Washington, DC, USA, 2009. IEEE Computer Society.

Mohammad Hammoudeh, Robert Newman, Christopher Dennett, and Sarah Mount, Application Domain Driven Data Visualisation Framework for Wireless Sensor Networks. The Sixth International Conference on Ubiquitous Intelligence and Computing (UIC-09), 2009.

Mohammad Hammoudeh, Robert Newman, and Sarah Mount. Modelling clustering in wireless sensor networks with synchronised hyperedge replacement. In *ICGT '08 Doctoral Symposium: Proceedings of the 4th international conference on Graph Transformations*, Electronic Communications of the EASST, 2009.

Mohammad Hammoudeh, Robert Newman, and Sarah Mount. An approach to data extraction and visualisation for wireless sensor networks.

In ICN 2009: *The Eighth International Conference on Networks*, Washington, DC, USA, 2009. IEEE Computer Society.

Mohammad Hammoudeh. Modelling clustering of sensor networks with synchronised hyperedge replacement (Extended Abstract). In *ICGT '08: Proceedings of the 4th international conference on Graph Transformations*, pages 490-492, Berlin, Heidelberg, 2008. Springer-Verlag.

Robert Newman and Mohammad Hammoudeh. Pennies from heaven: A retrospective on the use of wireless sensor networks for planetary exploration. In *AHS '08: Proceedings of the 2008 NASA/ESA Conference on Adaptive Hardware and Systems*, pages 263-270, Washington, DC, USA, 2008. IEEE Computer Society.

Mohammad Hammoudeh, James Shuttleworth, Robert Newman, and Sarah Mount. Experimental applications of hierarchical mapping services in wireless sensor networks. In *SENSORCOMM '08: Proceedings of the 2008 Second International Conference on Sensor Technologies and Applications*, pages 36-43, Washington, DC, USA, 2008. IEEE Computer Society.

Mohammad Hammoudeh, Alexander Kurz, and Elena Gaura. Mumhr: Multi-path, multi-hop hierarchical routing. In *SENSORCOMM '07: Proceedings of the 2007 International Conference on Sensor Technologies and Applications*, pages 140-145, Washington, DC, USA, 2007. IEEE Computer Society.

James Shuttleworth, Mohammad Hammoudeh, Elena Gaura, and Robert Newman. Experimental applications of mapping services in wireless sensor networks. In *Fourth International Conference on Networked Sensing Systems*, June 2007.

Bibliography

- Abdelzaher, T., He, T. and Stankovic, J. (2004), Feedback control of data aggregation in sensor networks, *in* ‘Decision and Control, 2004. CDC. 43rd IEEE Conference on’, Vol. 2, pp. 1490– 1495.
- ACM (2007), ‘The acm international symposium on mobile ad hoc networking and computing (mobi-hoc)’, <http://www.sigmobile.org/mobihoc>. [Online; accessed 2-November-2007].
- AG, T. (2008), ‘Oak smart usb devices’, http://www.toradex.com/En/Products/Oak_USB_Sensors. [Online; accessed 1-December-2008].
- Ahn, G.-S., Campbell, A. T., Veres, A. and Sun, L.-H. (2002), ‘Supporting service differentiation for real-time and best-effort traffic in stateless wireless ad hoc networks (swan)’, *IEEE Transactions on Mobile Computing* **1**(3), 192–207.
- Akyildiz, I., Vuran, M. and Akan, O. (2004), ‘On exploiting spatial and temporal correlation in wireless sensor networks’, *IEEE WiOpt’04, University of Cambridge* .
- Akyildiz, I., Weilian, S., Sankarasubramaniam, Y. and Cayirci, E. (2002), ‘A survey on sensor networks’, *Communications Magazine, IEEE* **40**, 102–114.
- Alazzawi, L. K., Elkateeb, A. M., Ramesh, A. and Aljuhar, W. (2008), Scalability analysis for wireless sensor networks routing protocols, *in* ‘AINAW

- '08: Proceedings of the 22nd International Conference on Advanced Information Networking and Applications - Workshops', IEEE Computer Society, Washington, DC, USA, pp. 139–144.
- Alzaid, H., Foo, E. and Nieto, J. G. (2008), Secure data aggregation in wireless sensor network: a survey, *in* 'AISC '08: Proceedings of the sixth Australasian conference on Information security', Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 93–105.
- Amis, A. D., Prakash, R., Huynh, D. and Vuong, T. (2000), Max-min d-cluster formation in wireless ad hoc networks, *in* 'INFOCOM', pp. 32–41.
- Anastasi, G., Conti, M., Francesco, M. D. and Passarella, A. (2009), 'Energy conservation in wireless sensor networks: A survey', *Ad Hoc Netw.* **7**(3), 537–568.
- Andrews, J. (1982), 'The history of topographical maps: Symbols, pictures and surveys', *Journal of Historical Geography* **8**(2), 218 – 219.
- Arboleda, L. and Nasser, N. (2007), Comparison of clustering algorithms and protocols for wireless sensor networks, *in* 'Proceedings of the Canadian Conference on Electrical and Computer Engineering', pp. 1787–1792.
- Babbitt, T. A., Morrell, C., Szymanski, B. K. and Branch, J. W. (2008a), 'Self-selecting reliable paths for wireless sensor network routing', *Comput. Commun.* **31**(16), 3799–3809.
- Babbitt, T. A., Morrell, C., Szymanski, B. K. and Branch, J. W. (2008b), 'Self-selecting reliable paths for wireless sensor network routing', *Comput. Commun.* **31**(16), 3799–3809.
- Bandyopadhyay, S. and Coyle, E. (2003), An energy efficient hierarchical clustering algorithm for wireless sensor networks, *in* 'Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom 2003)'.

- Banerjee, S. and Khuller, S. (2001), A clustering scheme for hierarchical control in multi-hop wireless networks, *in* 'INFOCOM', pp. 1028–1037.
- Basagni, S. (1999), Distributed clustering for ad hoc networks, *in* 'ISPAN '99: Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN '99)', IEEE Computer Society, Washington, DC, USA, p. 310.
- Becker, R. A., Eick, S. G. and Wilks, A. R. (1995), 'Visualizing network data', *IEEE Transactions on Visualization and Computer Graphics* **1**(1), 16–28.
- Beigl, M., Decker, C., Krohn, A., Riedel, T. and Zimmer, T. (2005), *in* 'Adjunct proceedings of the Ubicomp'.
- Berry, M. W. and Minser, K. S. (1999), 'Algorithm 798: high-dimensional interpolation using the modified shepard method', *ACM Trans. Math. Softw.* **25**(3), 353–366.
- Birdsong, L. and Helms, G. (2007), 'Visual universe-data into insight', <http://visualisation.notlong.com>. [Online; accessed 26-Jan-2009].
- Boonma, P. and Suzuki, J. (2007), 'Bisnet: A biologically-inspired middleware architecture for self-managing wireless sensor networks', *Comput. Netw.* **51**(16), 4599–4616.
- Bourbaki, N. (1966), *Elements of Mathematics: General Topology*, Addison-Wesley.
- Braunl, T. (2005), 'Visual universe-data into insight', <http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/java/Braunl/>. [Online; accessed 11-February-2009].

- Brodie, K. W., Asim, M. R. and Unsworth, K. (2005), 'Constrained visualization using the shepard interpolation family', *Comput. Graph. Forum* **24**(4), 809–820.
- Buisseret, D. (1992), *Monarchs, Ministers and Maps: The Emergence of Cartography as a Tool of Government in Early Modern Europe*, Chicago: University of Chicago Press.
- Burley, M., Bechkoum, K. and Pearce, G. (2006), A formative survey of geometric algebra for multivariate modelling, in 'UK Society for Modelling and Simulation', pp. 37–40.
- Caccamo, M., Zhang, L. Y., Sha, L. and Buttazzo, G. (2002), An implicit prioritized access protocol for wireless sensor networks, in 'RTSS '02: Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS'02)', IEEE Computer Society, Washington, DC, USA, p. 39.
- Cerpa, A. and Estrin, D. (2004), 'Ascent: Adaptive self-configuring sensor networks topologies', *IEEE Transactions on Mobile Computing* **3**(3), 272–285.
- Chan, H., Perrig, A. and Song, D. (2006), Secure hierarchical in-network aggregation in sensor networks, in 'CCS '06: Proceedings of the 13th ACM conference on Computer and communications security', ACM Press, New York, NY, USA, pp. 278–287.
- Chang, X. (1999), Network simulations with opnet, in 'WSC '99: Proceedings of the 31st conference on Winter simulation', ACM Press, New York, NY, USA, pp. 307–314.
- Chang, Y.-S., Lo, C.-J., Hsu, M.-T. and Huang, J.-H. (2006), Fault estimation and fault map construction on cluster-based wireless sensor network, in 'SUTC '06: Proceedings of the IEEE International Conference

- on Sensor Networks, Ubiquitous, and Trustworthy Computing - Vol 2 - Workshops', IEEE Computer Society, Washington, DC, USA, pp. 14–19.
- Chorzempa, M., Park, J.-M. and Eltoweissy, M. (2007), 'Key management for long-lived sensor networks in hostile environments', *Comput. Commun.* **30**(9), 1964–1979.
- Chu, D., Deshpande, A., Hellerstein, J. M. and Hong, W. (2006), Approximate data collection in sensor networks using probabilistic models, in 'ICDE '06: Proceedings of the 22nd International Conference on Data Engineering', IEEE Computer Society, Washington, DC, USA, p. 48.
- Chun-yan, S., Hua-zhong, Z. and Xiu-yang, Z. (2008), 'Clustering hierarchy tree routing algorithm based on leach', *Journal of Computer Applications* **28**(10), 2594 –.
- Curren, D. (2005), 'A survey of simulation in sensor networks', <http://www.cs.binghamton.edu/kang/teaching/cs580s/david.pdf>. [Online; accessed 27-August-2007].
- Dai, F. and Wu, J. (2004), 'Performance analysis of broadcast protocols in ad hoc networks based on self-pruning', *IEEE Trans. Parallel Distrib. Syst.* **15**(11), 1027–1040.
- Deng, J., Han, Y. S., Heinzelman, W. B. and Varshney, P. K. (2005), 'Scheduling sleeping nodes in high density cluster-based sensor networks', *Mob. Netw. Appl.* **10**(6), 825–835.
- Deshpande, A., Guestrin, C., Madden, S. R., Hellerstein, J. M. and Hong, W. (2004), Model-driven data acquisition in sensor networks, in 'VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases', VLDB Endowment, pp. 588–599.
- Downard, I. (2004), 'Simulating sensor networks in ns-2', *NRL/FR/5522-04-10073* .

- Dubois-Ferriere, H. and Estrin, D. (2004), Efficient and practical query scoping in sensor networks, *in* 'Mobile Ad-hoc and Sensor Systems', pp. 564 – 566.
- Dunn, J. and Martin, C. (2001), Terminology for frame relay benchmarking, *in* 'Internet informational RFC 3133'.
- Elmusrati, M., Jantti, R. and Koivo, H. (2005), Distributed sensor network data fusion using image processing, *in* 'Proceedings of Systems Communications', pp. 383–388.
- Estrin, D. (2007), Reflections on wireless sensing systems: From ecosystems to human systems, *in* 'Radio and Wireless Symposium', pp. 1 – 4.
- Extricom (2007), *Application Note - Wired and Wireless LAN Security*, Juniper Networks.
- Fang, Q., Gao, J. and Guibas, L. (2004), Locating and bypassing routing holes in sensor networks, *in* 'Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies', Vol. 4, pp. 2458 – 2468.
- FLIR Systems (2008), 'Thermacam p65', http://www.flir.com.hk/p65_print.htm. [Online; accessed 6-November-2008].
- Ganesan, D., Estrin, D. and Heidemann, J. (2003), 'Dimensions: why do we need a new data handling architecture for sensor networks?', *SIGCOMM Comput. Commun. Rev.* **33**(1), 143–148.
- Ganesan, D., Ratnasamy, S., Wang, H. and Estrin, D. (2004), Coping with irregular spatio-temporal sampling in sensor networks, *in* 'Sensor Networks SIGCOMM Comput. Commun. Rev.', pp. 125–130.

- Giridhar, A. and Kumar, P. (2006), Toward a theory of in-network computation in wireless sensor networks, *in* ‘Communications Magazine, IEEE’, Vol. 44, pp. 98 – 107.
- Girod, L., Ramanathan, N., Elson, J., Stathopoulos, T., Lukac, M. and Estrin, D. (2007a), ‘Emstar: A software environment for developing and deploying heterogeneous sensor-actuator networks’, *ACM Trans. Sen. Netw.* **3**(3), 13.
- Girod, L., Ramanathan, N., Elson, J., Stathopoulos, T., Lukac, M. and Estrin, D. (2007b), ‘Emstar: A software environment for developing and deploying heterogeneous sensor-actuator networks’, *ACM Trans. Sen. Netw.* **3**(3), 13.
- Goncalves, G. (2006), Analysis of interpolation errors in urban digital surface models created from lidar data, *in* M. Caetano and M. Painho, eds, ‘7th International Symposium on Spatial Accuracy Assessment in Natural Resources and Environmental Sciences’.
- Goodchild, M. and Kemp, K. (1990), *NCGIA Core Curriculum in GIS*, National Center for Geographic Information and Analysis.
- Greenstein, B., Kohler, E., Culler, D. and Estrin, D. (2004), Distributed techniques for area computation in sensor networks, *in* ‘29th Annual IEEE International Conference on Local Computer Networks (LCN’04)’, pp. 533 – 541.
- Gui, C. (2005), Routing Performance And Power Conservation In Ad Hoc and Sensor Networks, PhD thesis, Computer Science.
- Gumstix.com (2007), ‘Gumstix way small computing’, <http://www.gumstix.com>. [Online; accessed 26-March-2007].
- Hamann, B. and Joy, K. (2007), ‘Visualization methods for point data in space’, <http://www.citris-uc.org/research/projects/>

visualization_methods_for_point_data_in_space. [Online; accessed 05-June-2007].

Hammoudeh, M. (2006), Robust and energy efficient routing in wireless sensor networks, Master's thesis, University of Leicester.

Handy, M., Haase, M. and Timmermann, D. (2002), 'Low energy adaptive clustering hierarchy with deterministic cluster-head selection', *IEEE International Conference on Mobile and Wireless Communications Networks*.

Hartung, C., Han, R., Seielstad, C. and Holbrook, S. (2006), Firewxnet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments, *in* 'MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services', ACM, New York, NY, USA, pp. 28–41.

He, T., Krishnamurthy, S., Stankovic, J. A., Abdelzaher, T., Luo, L., Stoleru, R., Yan, T., Gu, L., Hui, J. and Krogh, B. (2004), Energy-efficient surveillance system using wireless sensor networks, *in* 'MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services', ACM, New York, NY, USA, pp. 270–283.

He, T., Stankovic, J. A., Lu, C. and Abdelzaher, T. (2003), Speed: A stateless protocol for real-time communication in sensor networks, *in* 'ICDCS '03: Proceedings of the 23rd International Conference on Distributed Computing Systems', IEEE Computer Society, Washington, DC, USA, p. 46.

Hefeeda, M. and Bagheri, M. (2007), Wireless sensor networks for early detection of forest fires, *in* 'International Workshop on Mobile Ad hoc and Sensor Systems for Global and Homeland Security (MASS-GHS 07)', Italy.

- Heinzelman, W. (2000), Application-specific protocol architectures for wireless networks, *in* 'PhD thesis, Massachusetts institute of technology'.
- Heinzelman, W., Chandrakasan, A. and Balakrishna, H. (2000), 'Energy-efficient communication protocol for wireless microsensor networks', *Proceedings of the 33rd International Conference on System Sciences*.
- Hellerstein, J., Hong, W., Madden, S. and Stanek, K. (2003), Beyond average: Towards sophisticated sensing with queries, *in* 'In Workshop on Information Processing In Sensor Networks (IPSN)'.
- Henninger, S., Lappala, K. and Raghavendran, A. (1995), An organizational learning approach to domain analysis, *in* 'International Conference on Software Engineering', pp. 95–104.
- Hertkorn, P. and Rudolph, S. (1998), Dimensional analysis in case-based reasoning, *in* S. I. fr Statik und Dynarnik der Loft-und Raumfahrtkonstruktionen, ed., 'International Workshop on Similarity Methods', pp. 163–178.
- Hertkorn, P. and Rudolph, S. (1999), From data to models: Synergies of a joint data mining and similarity theory approach, *in* 'SPIE Aerosense 1999 Conference On Data Mining and Knowledge Discovery', Orlando(USA).
- Hofierka, J., Parajka, J., Mitasova, H. and Mitas, L. (2002), 'Multivariate interpolation of precipitation using regularized spline with tension', *Transactions in GIS* **6**, 135–150.
- hp (2000), 'ipaq's', <http://www.hp.com/country/us/en/prodserv/handheld.html>. [Online; accessed 27-August-2007].
- Iyer, M. A., Watson, L. T. and Berry, M. W. (2006), Sheppack: a fortran 95 package for interpolation using the modified shepard algorithm, *in* 'ACM-

- SE 44: Proceedings of the 44th annual Southeast regional conference', ACM, New York, NY, USA, pp. 476–481.
- Jawhar, I., Mohamed, N., Shuaib, K. and Kesserwan, N. (2009), 'An efficient framework and networking protocol for linear wireless sensor networks', *Ad Hoc & Sensor Wireless Networks* **7**(1-2), 3–21.
- Joanes, D. and Gill, C. (1998), 'Comparing measures of sample skewness and kurtosis', *Journal of the Royal Statistical Society (Series D): The Statistician* **47**(1), 183–189.
- Johnson, D. B. and Maltz, D. A. (1996), Dynamic source routing in ad hoc wireless networks, in Imielinski and Korth, eds, 'Mobile Computing', Vol. 353, Kluwer Academic Publishers.
- Johnson, K. C. (2006), *Multidimensional Interpolation Methods*, KJ Innovation. <http://software.kjinnovation.com/InterpMethods.pdf>.
- Jr., F. C. and Bolstad, P. (1996), 'A comparison of spatial interpolation techniques in temperature estimation', http://www.ncgia.ucsb.edu/conf/SANTA_FE_CD-ROM/sf_papers/collins_fred/collins.html.
- Kang, H. and Li, X. (2006), Power-aware sensor selection in wireless sensor networks, in 'The 5th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)'.
- Karno, Z. (1992), 'Continuity of mappings over the union of subspaces', *Journal of Formalized Mathematics* **4**.
- Kačer, J. (2002), Discrete event simulations with j-sim, in 'PPPJ '02/IRE '02: Proceedings of the inaugural conference on the Principles and Practice of programming, 2002 and Proceedings of the second workshop on Intermediate representation engineering for virtual machines, 2002', National University of Ireland, Maynooth, County Kildare, Ireland, Ireland, pp. 13–18.

- Kim, J.-M., Park, S.-H., Han, Y.-J. and Chung, T.-M. (2008), Chef: Cluster head election mechanism using fuzzy logic in wireless sensor networks, Vol. 1, pp. 654–659.
- Kioskea.net (2008), ‘Propagation of radio waves (802.11)’, <http://en.kioskea.net/contents/wireless/wlpropa.php3>. [Online; accessed 28-Jan-2009].
- Kreylos, O. (2007), ‘Visualization of sensor network data’, <http://idav.ucdavis.edu/~okreylos/ResDev/SensorNetworks/index.html>. [Online; accessed 26-November-2007].
- Krivoruchko, K. (2004), *Using ArcGIS Geostatistical Analyst*, 9 edn, ESRI PRESS.
- Kulkarni, P., Ganesan, D. and Shenoy, P. (2005), ‘The case for multi-tier camera sensor networks’, *Proceedings of the 13th annual ACM international conference on Multimedia* pp. 229–238.
- Kumar, D., Aseri, T. C. and Patel, R. B. (2009), ‘Eehc: Energy efficient heterogeneous clustered scheme for wireless sensor networks’, *Comput. Commun.* **32**(4), 662–667.
- Kurkowski, S., Camp, T. and Colagrosso, M. (2005), Manet simulation studies: The current state and new simulation tools, Technical report, The Colorado School of Mines.
- Lazzaro, D. and Montefusco, L. (2002), ‘Radial basis functions for the multivariate interpolation of large scattered data sets’, *Journal of Computational and Applied Mathematics* pp. 521 – 536.
- Lee, H. Y., Seah, W. and Sun, P. (2006), Energy implications of clustering in heterogeneous wireless sensor networks an analytical view, pp. 1–5.

- Lee, S. and Chung, T. (2004), Data aggregation for wireless sensor networks using self-organizing map, *in* ‘Artificial Intelligence and Simulation’, Vol. 3397/2005, Springer Berlin / Heidelberg, pp. 508–517.
- Lee, S., Wolberg, G. and Shin, S. Y. (1997), ‘Scattered data interpolation with multilevel B-splines’, *IEEE Transactions on Visualization and Computer Graphics* **3**(3), 228–244.
- Levis, P., Lee, N., Welsh, M. and Culler, D. (2003), Tossim: accurate and scalable simulation of entire tinyos applications, *in* ‘SenSys ’03: Proceedings of the 1st international conference on Embedded networked sensor systems’, ACM Press, New York, NY, USA, pp. 126–137.
- Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E. and Culler, D. (2005), Tinyos: An operating system for sensor networks, *in* ‘In Ambient Intelligence’.
- Li, Z., Zhu, Q. and Gold, C. (2005), *Digital Terrain Modelling: principles and methodology*, CRC Press.
- Liarokapis, F., Newman, R., Goldsmith, D., Macan, L., Malone, G. and Shuttleworth, J. (2007), Sense-enabled mixed reality museum exhibitions, *in* ‘The 8th International Symposium on Virtual Reality, Archaeology and Cultural Heritage, VAST’.
- Lindsey, S., Raghavendra, C. and Sivalingam, K. (2001), Data gathering in sensor networks using the energy*delay metric, *in* ‘Proceedings of the IPDPS Workshop on Issues in Wireless Networks and Mobile Computing’.
- Liu, Y. and Li, M. (2007), Iso-map: Energy-efficient contour mapping in wireless sensor networks, *in* ‘ICDCS ’07: Proceedings of the 27th International Conference on Distributed Computing Systems’, IEEE Computer Society, Washington, DC, USA, p. 36.

- Liu, Y. and Seah, W. (2004), ‘A priority-based multi-path routing protocol for sensor networks’, *Personal, Indoor and Mobile Radio Communications* **1**, 216–220.
- Lou, W. and Wu, J. (2003), On reducing broadcast redundancy in ad hoc wireless networks, *in* ‘HICSS ’03: Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS’03) - Track 9’, IEEE Computer Society, Washington, DC, USA, p. 305.2.
- Lu, C., Blum, B. M., Abdelzaher, T. F., Stankovic, J. A. and He, T. (2002), Rap: A real-time communication architecture for large-scale wireless sensor networks, Technical report, Charlottesville, VA, USA.
- Madden, S. R., Franklin, M. J., Hellerstein, J. M. and Hong, W. (2005), ‘Tinydb: an acquisitional query processing system for sensor networks’, *ACM Trans. Database Syst.* **30**(1), 122–173.
- Manjeshwar, A. and Agrawal, D. (2001), A protocol for enhanced efficiency in wireless sensor networks, *in* ‘Proceedings of the 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing’.
- Manjeshwar, A. and Agrawal, D. (2002), Aptein: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks, *in* ‘Proceedings of the 2nd International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile computing’.
- Martinez, K., Basford, P., Ellul, J. and Spanton, R. (2009), Gumsense - a high power low power sensor node, *in* ‘6th European Conference on Wireless Sensor Networks.’.
- McCabe, C. (1998), ‘Grand canyon terrain’, Georgia Institute of Technology Large Geometric Models Archive. [Online; accessed 2-November-2007].

- Meng, X., Nandagopal, T., Li, L. and Lu, S. (2006), 'Contour maps: monitoring and diagnosis in sensor networks', *Comput. Netw.* **50**(15), 2820–2838.
- Minder, D., Grau, A. and Marrón, P. J. (2007), On group formation for self-adaptation in pervasive systems, in 'Autonomics '07: Proceedings of the 1st international conference on Autonomic computing and communication systems', ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, pp. 1–10.
- Mount, S. (2008), 'Dingo wireless sensor networks simulator', <http://code.google.com/p/dingo-wsn/>. [Online; accessed 26-October-2008].
- Mount, S., Newman, R. and Gaura, E. (2005), 'A simulation tool for system services in ad-hoc wireless sensor networks', *NSTI Nanotechnology Conference and Trade Show* **3**, 423.
- Mount, S., Newman, R., Gaura, E. and Kemp, J. (2006), Sensor: an algorithmic simulator for wireless sensor networks, in 'In Proceedings of Eurosensors 20', Vol. II, Gothenburg, Sweden, pp. 400–411.
- Muruganathan, S. D., Ma, D. C. F., Bhasin, R. I. and Fapojuwo, A. O. (2005), 'A centralized energy-efficient routing protocol for wireless sensor networks', *Communications Magazine, IEEE* **43**(3), 8–13.
- Naznin, M. and Nygard, K. (2006), Coverage in a heterogeneous sensor network, in 'Communications, Internet, and Information Technology'.
- NS-2 (2007), 'The network simulator', <http://www.isi.edu/nsnam/ns/>. [Online; accessed 1-November-2007].
- Nugent, R., Munakata, R., Chin, A., Coelho, R. and Puig-Suari, J. (2008), The cubesat: The picosatellite standard for research and education, in 'AIAA SPACE 2008 Conference and Exposition'.

- OPENET Technologies Inc (2007), ‘The acm international symposium on mobile ad hoc networking and computing (mobi-hoc)’, <http://www.opnet.com/>. [Online; accessed 2-August-2007].
- Pai, S. (2007), ‘Efficient visualization of streaming sensor network data using approximation technique’, <http://www.cse.uta.edu/news/seminars/ShowAbstract.asp?pageVer=grph&type=MS&id=385>. [Online; accessed 05-June-2007].
- Pang, A., Furman, J. and Nuss, W. (1994), ‘Data quality issues in visualization’, *In SPIE Proceedings on Visual Data Exploration and Analysis* **2178**, 12–23.
- Park, S., Savvides, A. and Srivastava, M. B. (2000), Sensorsim: a simulation framework for sensor networks, *in* ‘MSWIM ’00: Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems’, ACM Press, New York, NY, USA, pp. 104–111.
- Park, S. W., Linsen, L., Kreylos, O., Owens, J. D. and Hamann, B. (2005), A framework for real-time volume visualization of streaming scattered data, *in* ‘Proceedings of the Tenth International all Workshop on Vision, Modeling, and Visualization’, pp. 225–232.
- PDKB (2008), ‘Artificial intelligence knowledge bank of commonsense rules and facts’, <http://sourceforge.net/projects/pdkb/>. [Online; accessed 27-May-2008].
- Perkins, C. E. and Royer, E. M. (1999), Ad hoc on-demand distance vector routing, *in* ‘Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)’.

- Perrone, L. F. and Nicol, D. M. (2002), A scalable simulator for tinyos applications, *in* ‘Simulation Conference, 2002. Proceedings of the Winter’, Vol. 1, pp. 679 – 687.
- Pietro, R. D., Ma, D., Soriente, C. and Tsudik, G. (2008), Posh: Proactive co-operative self-healing in unattended wireless sensor networks, *in* ‘SRDS ’08: Proceedings of the 2008 Symposium on Reliable Distributed Systems’, IEEE Computer Society, Washington, DC, USA, pp. 185–194.
- Pottie, G. J. and Kaiser, W. J. (2000), ‘Wireless integrated network sensors’, *Commun. ACM* **43**(5), 51–58.
- Raweb, I. (2004), ‘Resource sharing in wireless and sensor networks’, <http://ralyx.inria.fr/2004/Raweb/mascotte/uid30.html>. [Online; accessed 06-June-2007].
- Renka, R. J. and Brown, R. (1999*a*), ‘Algorithm 791: Tshep2d: cosine series shepard method for bivariate interpolation of scattered data’, *ACM Trans. Math. Softw.* **25**(1), 74–77.
- Renka, R. J. and Brown, R. (1999*b*), ‘Algorithm 792: accuracy test of acm algorithms for interpolation of scattered data in the plane’, *ACM Trans. Math. Softw.* **25**(1), 78–94.
- Ruprecht, D. and Müller, H. (1994), A framework for generalized scattered data interpolation, *in* M. Göbel, H. Müller and B. Urban, eds, ‘Visualization in Scientific Computing’, Springer-Verlag Wien, pp. 72–86.
- Ruprecht, D. and Muller, H. (1995), Image warping with scattered data interpolation, *in* ‘IEEE Computer Graphics’, Vol. 15, pp. 37–43.
- Ruprecht, D., Nagel, R. and Müller, H. (1995), ‘Spatial free-form deformation with scattered data interpolation methods’, *Computers and Graphics* **19**(1), 63–71.

- Shepard, D. (1968), A two-dimensional interpolation function for irregularly-spaced data, *in* 'Proceedings of the 1968 23rd ACM national conference', ACM Press, pp. 517–524.
- Shuttleworth, J., Gaura, E. and Newman, R. M. (2006), Surface reconstruction: Hardware requirements of a som implementation, *in* 'Proceedings of the ACM Workshop on Real-World Wireless Sensor Networks (REAL-WSN'06)'.
- Shuttleworth, J., Hammoudeh, M., Gaura, E. and Newman, R. (2007), Experimental applications of mapping services in wireless sensor networks, *in* 'Fourth International Conference on Networked Sensing Systems'.
- S.Lindsey and Raghavendra, C. (2002), Pegasus: Power efficient gathering in sensor information systems, *in* 'Proceedings of the IEEE Aerospace Conference'.
- Smaragdakis, G., Matta, I. and Bestavros, A. (2004), Sep: A stable election protocol for clustered heterogeneous wireless sensor networks, *in* 'Second International Workshop on Sensor and Actor Network Protocols and Applications (SANPA 2004)'.
- Sobeih, A., Chen, W.-P., Hou, J. C., Kung, L.-C., Li, N., Lim, H., Tyan, H.-Y. and Zhang, H. (2005), J-sim: A simulation environment for wireless sensor networks, *in* 'ANSS '05: Proceedings of the 38th annual Symposium on Simulation', IEEE Computer Society, Washington, DC, USA, pp. 175–187.
- Solis, I. and Obraczka, K. (2003), In-network aggregation trade-offs for data collection in wireless sensor networks, Technical report, INRG Technical Report 102.

- Sugihara, R. and Chien, A. (2005), Accuracy-aware data modeling in sensor networks, *in* ‘3rd international conference on Embedded networked sensor systems’, pp. 282–283.
- Sun, P., Seah, W. K. and Lee, P. W. (2007), Efficient data delivery with packet cloning for underwater sensor networks, *in* ‘Symposium on Underwater Technology and Workshop on Scientific Use of Submarine Cables and Related Technologies’, pp. 34–41.
- Sundresh, S., Kim, W. and Agha, G. (2004), Sens: A sensor, environment and network simulator, *in* ‘The 37th Annual Simulation Symposium (ANSS37)’.
- Tran Quang, V. and Miyoshi, T. (2008), ‘Adaptive Routing Protocol with Energy Efficiency and Event Clustering for Wireless Sensor Networks’, *IEICE Trans Commun* **E91-B**(9), 2795–2805.
URL: <http://ietcom.oxfordjournals.org/cgi/content/abstract/E91-B/9/2795>
- Tubaishat, M., Yin, J., Panja, B. and Madria, S. (2004), ‘A secure hierarchical model for sensor network’, *SIGMOD Rec.* **33**(1), 7–13.
- Turau, V. and Weyer, C. (2009), ‘Fault tolerance in wireless sensor networks through self-stabilisation’, *Int. J. Commun. Netw. Distrib. Syst.* **2**(1), 78–98.
- Tynan, R., O’Hare, G., Marsh, D. and O’Kane, D. (2005), Interpolation for wireless sensor network power management, *in* ‘International Workshop on Wireless and Sensor Networks (WSNET-05)’, IEEE Press.
- Ulmer, C. (2007), ‘Wireless sensor probe networks-sensorsimii’, <http://www.craigulmer.com/research/sensorsimii/>. [Online; accessed 18-August-2007].

- Vidhyapriya, R. and Yu, P. T. V. (2007), 'Energy aware routing for wireless sensor networks', *INFOCOMP Journal of Computer Science* **6**(3), 07–14.
- Virrankoski, R. Savvidees, A. (2005), Tasc: topology adaptive spatial clustering for sensor networks, *in* 'Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on'.
- Wan, J., Chen, W., Xu, X. and Fang, M. (2008), An efficient self-healing scheme for wireless sensor networks, *in* 'FGCN '08: Proceedings of the 2008 Second International Conference on Future Generation Communication and Networking', IEEE Computer Society, Washington, DC, USA, pp. 98–101.
- Wang, Y.-H., Tsai, C.-H., Mao, H.-J. and Huang, K.-F. (2006), An energy-efficient hierarchical multiple-choice routing path protocol for wireless sensor networks, *in* 'SUTC '06: Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing -Vol 1 (SUTC'06)', IEEE Computer Society, Washington, DC, USA, pp. 570–571.
- Wei, X., Ya-qing, T., Hua, X. and Yu-wen, M. (2008), 'Hierarchical routing protocol to handle topological variety for wireless sensor network', *Chinese Journal of Sensors and Actuators* **21**(9), 1635 – 1639.
- Wood, A. D., Stankovic, J. A. and Son, S. H. (2003), Jam: A jammed-area mapping service for sensor networks, *in* 'RTSS '03: Proceedings of the 24th IEEE International Real-Time Systems Symposium', IEEE Computer Society, Washington, DC, USA, p. 286.
- Wu, C., Lee, K. and Chung, Y. (2006), A delaunay triangulation based method for wireless sensor network deployment, *in* '12th International Conference on Parallel and Distributed Systems', Vol. 1, pp. 253–260.

- Wu, Y., Fahmy, S. and Shroff, N. B. (2007), Energy efficient sleep/wake scheduling for multi-hop sensor networks: Non-convexity and approximation algorithm, *in* 'INFOCOM 2007 26th IEEE International Conference on Computer Communications', pp. 1568–1576.
- xbow (2007), 'Mica mote', <http://www.xbow.com/>. [Online; accessed 19-August-2007].
- Xue, W., Luo, Q., Chen, L. and Liu, Y. (2006), Contour map matching for event detection in sensor networks, *in* 'SIGMOD '06: Proceedings of the 2006 ACM SIGMOD international conference on Management of data', ACM Press, New York, NY, USA, pp. 145–156.
- Yang, C., Kao, S., Lee, F. and Hung, P. (2004), Twelve different interpolation methods: A case study of surfer 8.0, *in* 'Geo-Imagery Bridging Continents', p. 778.
- Yang, Y., Wu, H. and Zhuang, W. (2006), Mester: minimum energy spanning tree for efficient routing in wireless sensor networks, *in* 'QShine '06: Proceedings of the 3rd international conference on Quality of service in heterogeneous wired/wireless networks', ACM, New York, NY, USA, p. 17.
- Ye, M., Li, C., Chen, G. and Wu, J. (2005), Eecs: An energy efficient clustering scheme in wireless sensor networks, *in* 'Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International', pp. 535– 540.
- Younis, O. and Fahmy, S. (2004), 'Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks', *IEEE Transactions on Mobile Computing* **3**(4), 366–379.
- Yu, Y. (1999), Surface reconstruction from unorganized points using self-organizing neural networks, *in* 'IEEE Visualization '99', pp. 61–64.

- Zeng, X., Bagrodia, R. and Gerla, M. (1998), Glomosim: a library for parallel simulation of large-scale wireless networks, *in* ‘PADS ’98: Proceedings of the twelfth workshop on Parallel and distributed simulation’, IEEE Computer Society, Washington, DC, USA, pp. 154–161.
- Zhao, Y. J., Govindan, R. and Estrin, D. (2002), Residual energy scan for monitoring sensor networks, *in* ‘Wireless Communications and Networking Conference’, pp. 356–362.

Appendix A

A Survey on Wireless Sensor Networks Simulators

A.1 SensorSim

SensorSim (Park et al., 2000) builds up on the NS-2 simulator providing additional capabilities for modelling wireless sensor networks. The main features of this platform are: power and communication protocol models; sensing channel and sensor models; scenario generation; and support for hybrid simulations. The public release of the SensorSim suite of tools has been withdrawn due to its unfinished nature and the inability of authors to provide the needed level of support.

Georgia Tech SensorSimII (Ulmer, 2007) is written in a modular style which results in organising a sensor node into three components: application, network, and link. The work in SensorSimII may be divided into two areas: the simulator core and the visualisation tools. The simulator core essentially manages an array of independent sensor nodes throughout time. While the visualisation tools give a visual representation about individual node state and communication among nodes.

Both SensorSim projects are open source and free to use. Away from

SensorSim's power modules, neither simulator takes into account sensor nodes limited resources which limit the realism of simulation results. Moreover, it is not always required by the wireless sensor network to validate the functional correctness and provide performance guarantees. As opposed to SensorSim's simulation mechanism requirements, in many wireless sensor network applications, the complete protocol stack is not needed to simulate the expected behaviour. This makes the SensorSim platform complex and difficult to use.

A.2 TOSSIM

There are platforms specifically designed to simulate Wireless Sensor Networks, such as TOSSIM (Levis et al., 2003) which is a part of the TinyOS development efforts (Levis et al., 2005). TOSSIM is a discrete-event simulator for TinyOS applications (xbow, 2007). It aims to assist TinyOS application development and debugging by compiling applications into the TOSSIM framework which runs on a PC instead of compiling them for a mote. Using the TOSSIM framework, programs can be directly targeted to motes without modification. This gives users a bigger margin to debug, test, and analyse algorithms in a controlled and repeatable environment. In TOSSIM, all nodes share the exact same code image, simulated at bit granularity, and assuming static node connectivity is known in advance. Therefore, TOSSIM is more of a TinyOS emulator than a general wireless sensor network simulator. It focuses on simulating TinyOS rather than simulating the real world. This has the advantage that the developed algorithms can be tested on a target platform. However, this may place some restrictions of the target platform on the simulation. TOSSIM is not always the right simulation solution; like any simulation, it makes several assumptions about the target hardware platform, focusing on making some behaviour accurate while simplifying others (Levis et al., 2003). TOSSIM

can be used as a tool for absolute evaluation of some causes of the behaviour observed in the real-world network deployments.

A.3 TOSSF

TOSSF (Perrone and Nicol, 2002) is a simulation framework that compiles a TinyOS application into the SWAN (Ahn et al., 2002) simulation framework. It can be viewed as an improvement over TOSSIM with a primary focus on scalability. It allows simulation of a heterogeneous collection of sensor nodes and a dynamic network topology. TOSSF suffers from potentially long test-debug cycles because it does not provide a scripting framework for experimentation. Although it enables development of custom environmental models, the absence of a scripting framework requires those models to be compiled into the simulation framework. Given that both of these simulators are tightly coupled with TinyOS, they may be unsuitable for early prototyping or developing portable wireless sensor network applications.

A.4 GloMoSim

GloMoSim (Zeng et al., 1998) is a scalable simulation environment for wireless and wired network systems. Its parallel discrete-event design distinguishes it from most other sensor network simulators. Though it is a general network simulator, GloMoSim currently supports protocols designed purely for wireless networks. GloMoSim is built using a layered approach similar to the seven layer network architecture of the OSI model. It uses standard APIs between different simulation layers to allow rapid integration of models developed at different layers, possibly by different users.

As in NS-2, GloMoSim uses an object-oriented approach, however for scalability purposes; each object is responsible for running one layer in the protocol stack of every node. This design strategy helps to divide the

overhead management of a large-scale network. GloMoSim has been found to be effective for simulating IP networks, but it is not capable of simulating sensor networks accurately (Curren, 2005). Moreover, GloMoSim does not support phenomena occurring outside of the simulation environment, all events must be gathered from neighbouring nodes in the network. Finally, GloMoSim stopped releasing updates in 2000 and released a commercial product called QualNet.

A.5 OPNET

OPNET (Chang, 1999) is a further discrete event, object oriented, general purpose network simulator. The engine of OPNET is a finite state machine model in combination with an analytical model. It uses a hierarchical model to define each characteristic of the system. The top hierarchy level contains the network model, where the topology is designed. The second level defines the data flow models. The third level is the process editor which handles control flow models defined in the second level. Finally, a parameter editor is included to support the three higher levels. The hierarchical models result in event queue for the discrete event simulation engine and a set of entities that handle the events. Each entity represents a node which consists of a finite state machine which processes the events during simulation.

Unlike NS-2 and GloMoSim, OPNET supports modelling sensor-specific hardware, such as physical-link transceivers and antennas. It also enables users to define custom packet formats. An attractive feature of OPNET is its capability of recording a large set of user defined results. Furthermore, the GUI (Graphical User Interface), along with the considerable amount of documentation and study cases that come along with the license are another attractive feature of the simulator. This GUI interface can also be used to model, graph, and animate the resulting output. The network operator is provided with editors that are required to simplify the different levels

of modelling. Though model parameters can be changed, the simulation accuracy is influenced because OPNET is not open source software.

Similar to NS-2, the object-oriented design of OPNET causes scalability problems. It does not have a high number of protocols publicly available possibly because of the license constraints. Finally, OPNET is only available in commercial form.

The second class of simulators are application-oriented simulators, including EmStar (Girod et al., 2007b), SENS (Sundresh et al., 2004), J-Sim (Sobeih et al., 2005), and SenSor (Mount et al., 2006).

A.6 EmStar

EmStar (Girod et al., 2007b) is a component based, discrete-event framework that offers a range of run-time environments, from pure simulation, distributed deployment on iPAQs (hp, 2000), to a hybrid simulation mode similar to SensorSim. Emstar supports the use of simulation in the early stages of design and development by providing a range of simulated sensor network components, including radios, which provide the same interfaces as actual components. It supports hybrid mode with some actual components and some simulated components, and full native mode with no simulated components. As in TOSSIM, EmStar uses the same source code that runs at each of these levels to run on actual sensors. Amongst other simulators, such as TOSSIM, EmStar provides an option to interface with actual hardware while running a simulation. EmStar is compatible with two different types of node hardware. It can be used to develop software for Mica2 motes and it also offers support for developing software for iPAQ based microservers. The development cycle is the same for both hardware platforms. The next step in the development cycle following the simulation is data replay. In this model, EmStar uses data collected from actual sensors in order to run its simulation. Leading directly from this, Emstar uses the half-simulation

methodology similar to SensorSim's, where the software is running on a host machine and interfacing with a real physical communication channels. The final step in the development cycle is deployment.

EmStar combines many of the good features of other wireless sensor networks simulators. Its component based design allows for fair scalability. Moreover, each aspect of the network can be logically fine-tuned due to its development cycle design. Because it targets a particular platform, many protocols are already available to be used. At the deployment step in the development cycle, only the configuration files have to be designed. This potentially adds constraints on the user through either ensure that the hardware configuration being used matches the existing configuration file, or he must write his own files.

The main goal of Emstar is to reduce the design complexity, enabling work to be shared and reused, and simplifying and speeding the design of new sensor network applications. While not as efficient and fast as other frameworks like TOSSIM, Emstar provides a simple environmental model and network medium in which to design, develop and deploy heterogeneous sensor network applications. When used as a migration platform from code to real sensor environment, the environment model may be sufficient for most developers. Another drawback of Emstar is that the simulator supports only the code for the types of nodes that it is designed to work with.

A.7 SENS

SENS (Sundresh et al., 2004) is a customisable component-based simulator for wireless sensor network applications. It consists of interchangeable and extensible components for applications, network communication, and the physical environment. In SENS, each node is partitioned into four main components: application, simulates the software application of the sensor node; network, handles incoming and outgoing packets; physical,

reads sensed information; and environment, network propagation characteristics. Multiple different component implementations offer varying degrees of realism. For example, users can choose between various application-specific environments with different signal propagation characteristics. As in TOSSIM, SENS source code can be ported directly into actual sensor nodes, enabling application portability. Moreover, it provides a power module for development of dependable applications.

SENS defines three network models that can be used. The first successfully forwards packets to all neighbours, the second delivers with a chance of loss based on a fixed probability, and the third considers the chance of collision at each node. The physical component includes the non-network hardware for the sensor such as the power, sensors, and actuators. At a lower level, the environment component models the physical phenomena and the layout. The layout model includes different types of surfaces, each affecting radio and sound propagation in a different way.

The drawback of SENS is that it is less customisable than many other simulators, providing no chance to alter the MAC protocol, along with other low level network protocols. SENS uses one of the most sophisticated environmental models and implements the use of sensors well. However, the only measurable phenomenon is sound.

A.8 J-Sim

J-Sim (Sobeih et al., 2005) is a component-based discrete event simulator built in Java and modelled after NS-2. The design of this simulator aims at solving many of the shortcomings of comparable object-oriented simulators like NS-2. J-Sim uses the concept of components instead of the concept of having an object for each individual node. J-Sim uses three top level components: the target node which produces stimuli, the sensor node that reacts to the stimuli, and the sink node which is the ultimate destination

for stimuli reporting. Each component is broken into parts and modelled differently within the simulator; this eases the use of different protocols in different simulation runs.

The authors of J-Sim claim that it has several advantages over NS-2 and other simulators. First its component based architecture scales better than the object oriented model used by NS-2 and other simulators. Second, J-Sim has an improved energy model and the ability to simulate the use of sensors for phenomena detection. Like SensorSim, there is support for using the simulation code for real hardware sensors. However, J-Sim is comparatively complicated to use. While no more complicated than NS-2, the latter simulator is more popular and accepted in the sensor network research community and more community support is available, therefore, more people are keen to spend the time to learn how to use it.

Though is scalable, J-Sim has a set of inefficiencies. First, there is unnecessary overhead in the intercommunication model. The second problem is inherited by most sensor networks simulators that are built on top of general purpose simulators, 802.11 is the only MAC protocol that can be used in J-Sim. Finally, Java is possibly less efficient than many other languages.

Appendix B

A Survey on Cluster-based Routing Algorithms

B.1 LEACH

LEACH (Heinzelman et al., 2000) is one of the most promising routing algorithms for sensor networks. It is a clustering-based protocol that utilises randomised rotation of the cluster head role to evenly distribute the energy load among the sensors in the network. It guarantees that the energy load is well distributed by dynamically created clusters. The operation of LEACH is split into two phases, the setup phase and the steady-state phase. To minimise overhead, the duration of the steady-state phase is longer than the set-up phase (Heinzelman et al., 2000). During the set-up phase, the clusters are created and cluster heads are elected. This election is made by every node choosing a random number between 0 and 1. The sensor node becomes a cluster head if this random number is less than the threshold $T(n)$ set as:

$$T(n) = \begin{cases} P / (1 - P \times (r \bmod \frac{1}{P})) & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases}$$

where P is the desired percentage of cluster heads, r is the current round and G is the set of nodes that have not been selected as a cluster head in the last $1/P$ rounds. The desired percentage of cluster heads was found to be 5% of the total number of nodes in the network (Heinzelman et al., 2000). Note that 0 cluster heads and 100% cluster heads is equivalent to one-hop communication.

After cluster heads are chosen, they broadcast an advertisement message to the entire network declaring that they are the new cluster heads. Every node receiving the advertisement decides to which cluster they wish to belong based on the signal strength of the received message. The sensor node sends a message to register with the cluster head of their choice. Based on a TDMA approach, the cluster head assigns each node registered in its cluster a time slot to send its data. This requires that every node must support TDMA. The cluster head keeps a list of all nodes in the cluster to inform them of their TDMA schedule.

During the steady-state phase, sensor nodes can start transmitting data to their respective cluster head. The cluster head applies aggregation functions to compress the data before transmission to the sink. After a predetermined period of time spent on the steady-state phase, the network enters the set-up phase again and starts a new round of creating clusters.

LEACH is well-suited for applications where constant monitoring is needed and data collection occurs periodically to a centralised location (Lee and Chung, 2004). It increases network lifetime in two ways. First, load is distributed to all nodes but not all at the same time. Second, there is lossless aggregation of data by the cluster heads. The protocol is powerful and simple since nodes do not require global knowledge or location information to create clusters. LEACH is able to increase the network lifetime and it achieves a more than 7-fold reduction in energy dissipation compared to direct communication (Heinzelman et al., 2000).

Despite the significant overall energy savings, the assumptions made by

the protocol raise a number of issues:

1. LEACH assumes that all nodes begin with the same amount of energy and that the amount of energy a cluster head consumes is more than that of a non-cluster node. It also assumes that the amount of energy consumed by cluster heads in every cluster round is constant. This assumption is not realistic. Furthermore, making adjustments for differences in energy consumption causes LEACH to be less efficient.
2. LEACH assumes that all nodes can communicate with each other and are able to reach the sink. Therefore, it is only suitable for small size networks.
3. LEACH requires all nodes to be continuously listening. This is not realistic in random node deployment networks, for example, where cluster heads could be located at the edge of the network.
4. LEACH assumes that all nodes have data to send and so assign a time slot for a node even though some nodes might not have data to transmit.
5. LEACH assumes that all nearby nodes have correlated data which is not always true.
6. Finally, there is no mechanism to ensure that the elected cluster heads (P) will be uniformly distributed over the network. Hence, there is the possibility that all cluster- heads will be concentrated in one part of the network.

LEACH-C (LEACH Centralised) (Heinzelman, 2000) has been developed out of LEACH. It uses a central algorithm to form clusters. During the set-up phase, the sink receives information from each node in the network about its current location and energy level. The sink then runs a centralised cluster formation algorithm to determine the clusters for that round. This

algorithm is not robust since it requires location information for all sensors in the network. It also requires communication with the base station to get optimal clusters.

A number of enhancements over LEACH have been proposed previously (Heinzelman, 2000; Lindsey et al., 2001; Manjeshwar and Agrawal, 2001, 2002; S.Lindsey and Raghavendra, 2002; Smaragdakis et al., 2004; Ye et al., 2005).

B.2 PEGASIS

S.Lindsey and Raghavendra (2002) devised a protocol called Power Efficient Gathering in Sensor Information Systems (PEGASIS) that is an improvement over LEACH. As opposed to LEACH, PEGASIS has no clusters, instead it creates chains from the farthest sensor node from the sink so that each node communicates only with their closest neighbours and only one node is selected from the chain to communicate with the sink.

In PEGASIS, each chain has a leader for collecting data along both sides of the chain and transmitting the final aggregated packet to the sink. Also, each node takes turns to serve as a leader to distributed energy load evenly among all nodes. By minimising the communication distances between neighbouring nodes on the chain, PEGASIS reduces the total transmission distance and achieves longer network lifetime than LEACH for different network topologies and sizes. Nevertheless, PEGASIS has a number of drawbacks (Hammoudeh, 2006):

1. It requires dynamic adjustment of network topology to route data, which introduces significant overhead
2. It requires location information. Since the sensor nodes might be mobile, acquiring node location information is not an easy task in sensor networks. Although adding GPS could be an alternative, it is

an unpractical solution as it may increase the size and price of sensor nodes.

3. Similar to LEACH, PEGASIS assumes that all nodes can communicate with the sink directly
4. The head of the chain can become a bottleneck and cause excessive transmission delays
5. It assumes that all nodes start with the same level of energy and consumption rates are equal

Hierarchical-PEGASIS (Lindsey et al., 2001) extends PEGASIS by introducing a hierarchy in the network topology. It aims to reduce transmission delays to the sink and proposes a data gathering scheme that balances the energy and delay cost. Although Hierarchical-PEGASIS avoids the clustering overhead, it still requires dynamic topological adjustments.

B.3 TEEN and APTEEN

Manjeshwar and Agrawal (2001) implemented the Threshold sensitive Energy Efficient Protocol (TEEN) that utilises a hierarchical approach along with a data centric mechanism (Manjeshwar and Agrawal, 2001). The same authors also developed the AdaPtive Threshold-sensitive Energy Efficient sensor Network protocol (APTEEN) (Manjeshwar and Agrawal, 2002), which enhances TEEN by capturing periodic data collection and reacting to time critical events. These protocols were proposed for time-critical applications.

In TEEN, sensor nodes sense the environment continuously, but the data transmission is done less frequently. It allows the user to control the trade-off between energy efficiency and data accuracy using threshold values. The cluster head sends its members a hard threshold, which is the threshold

value of the sensed attribute and a soft threshold, which is the change in the value of the sensed attribute that triggers the node to transmit. Accordingly the hard threshold reduces the number of transmissions by allowing the nodes to transmit only when the sensed attribute is within the range of interest. If the next sample differs from the previous one by at least the soft threshold, this information should be transmitted back to the base station. A smaller value of the soft threshold gives a more accurate picture of the network, at the expense of increased energy consumption. At each cluster formation round, new values for the above parameters are broadcast.

As in TEEN, the nodes in APTEEN sense the environment continuously, and only those data values at, or beyond, the hard threshold are transmitted. APTEEN combines both reactive and proactive policies by introducing a new parameter called count time. The count time is defined as the maximum time period between two successive reports sent by a node. Sensing nodes only transmit when the sensed value changes by an amount equal to or greater than the soft threshold. If a node does not send data for a time period equal to the count time, it is forced to transmit the current sensed value. APTEEN offers more flexibility than TEEN by allowing the user to set the count-time interval, and control the threshold values for the energy consumption by changing the count time as well as the threshold values. APTEEN implements hybrid networks more efficiently than TEEN and LEACH by using a modified TDMA schedule.

APTEEN performance was demonstrated to be between LEACH and TEEN in terms of energy dissipation and network lifetime. TEEN gives the best performance since it decreases the number of transmissions. Both TEEN and APTEEN protocols require additional traffic control to continually update the threshold values and complexity of forming clusters in multiple levels, implementing threshold-based functions and dealing with attribute-based naming of queries.

B.4 SEP

Smaragdakis et al. (2004) address the issue of heterogeneity of nodes in terms of their energy. SEP is a heterogeneous-aware protocol that is based on the random selection of cluster heads weighted according to the remaining node energy to prolong the time interval before the death of the first node. Every sensor node in a heterogeneous two-level hierarchical network independently elects itself as a cluster head based on its initial energy relative to that of other nodes. Nodes do not require any global knowledge of energy at every election round. Unlike LEACH, SEP is dynamic in that it does not assume any prior distribution of the different levels of energy in the sensor nodes. Furthermore, SEP was found to be more flexible than LEACH in sensibly consuming the extra energy of the more powerful nodes.

This approach addresses the problem of varying energy levels and consumption rates but still assumes that the sink can be reached directly by all nodes. In addition, it only considers the nodes available energy in the election of cluster heads. Finally, it requires knowledge about the energy levels of other nodes in the network to weight the cluster head election probabilities.

B.5 MESTER

Unlike previous work targeting at maximising energy efficiency and network lifetime, Yang et al. (2006) proposed Minimum Energy Spanning Tree for Efficient Routing (MESTER) for maintaining a high quality in data collection for as long as possible. Compared with the existing Minimum Spanning Tree (MST), MESTER can achieve comparable but more balanced performance at a lower complexity. Similar to PEGASIS, the MESTER routing scheme adopts centralised algorithms and requires the sink to take control of managing the network topology and calculating the routing path and time schedule for data collection. MESTER operation consists of three phases:

1. MST calculation: A new MST is calculated by the sink when the network is just deployed and initialised and every time the network topology changes. Firstly, the sink randomly selects a node as the root. The corresponding MST is then derived by using communication distance as link weight. Starting from the root, the MST tree grows by connecting to the closest alive neighbouring node one by one until all alive nodes are sequentially added to the tree.
2. Communication pair and time schedule: After the leader is selected, simultaneous communication pairs in each time slot are scheduled along the MST for transmitting data packets towards the leader. After receiving the data packets from all its connected downstream nodes, an intermediate node performs data aggregation and transmits the aggregated packet to its upstream node. When two or more downstream nodes are simultaneously ready to transmit their data packets to the same upstream node, an arbitrary transmission sequence is determined by the sink and a TDMA approach is used to schedule the corresponding packet transmissions into different time slots to avoid collisions.
3. Data Transmission: The time schedule generated in phase 2 is broadcast to all of the alive sensor nodes to prepare them before each data collection round.

MESTER decouples the task of energy load balancing from the calculation of MST, and thus gives simpler, more robust, and more flexible algorithms. It also simply uses communication distance, rather than sophisticated energy cost functions as link weight in the calculation of MST. However, MESTER has a set of draw backs. First, nodes require location information and the location of all nodes should be known to the sink. Second, it is unsuitable for networks with frequent topological changes because of the overhead of calculating a new MST each time such a change occurs.

Third, as in LEACH, it assumes that every alive node has a data packet to transmit in each round of data collection. Fourth, it requires time synchronisation which is realised by receiving specific signals from either the GPS or the Sink. Fifth, as in PEGASIS the leader node can become a bottleneck and cause excessive transmission delays. Sixth, it is a centralised algorithm which makes it less favourable as the network grows larger. Furthermore, it requires that an intelligent sink be deployed with the nodes, which may not always be possible.

B.6 A secure hierarchical model

Tubaishat et al. (2004) proposed an energy-efficient multi-hop hierarchical routing protocol. Also, they devised a Secure Routing Protocol for Sensor Networks (SRPSN) that provides protection against diverse classes of attacks and guarantees packet delivery to the sink in the presence of adversaries.

The protocol builds the communication hierarchically levels based on the number of neighbours of each node. If a node has a larger number of neighbours, then it will be assigned a higher level. After a node discovers its number of neighbours (NBR) by a *ping* message, it exchanges its ID and NBRs with all of its neighbours to build clusters. Nodes which have the highest NBRs become cluster heads. As in MuMHR, cluster heads perform data aggregation to achieve energy savings. A node that is connected to two or more cluster heads moves to upper layers of the hierarchy and becomes a *root*. However, nodes at higher level may suffer from energy depletion and communication bottlenecks. To reduce these undesirable effects, cluster heads were programmed to increase data filtering and aggregation. The increased computation at cluster heads may lead to the same undesirable consequences that the algorithm is trying to avoid at the root nodes.

This routing algorithm requires that all sensor nodes are equipped with

a position-determining system, e.g. GPS, which is used by nodes to register with the nearest cluster head. Using the location information, an intermediate sensor node forwards a received packet to the neighbour that is closest to the location of the sink node as the next hop node in the communication path.

In the SRPSN protocol, each sensor node shares a secret key with the sink. The sink maintains a table of (id, key) pairs for all nodes. The protocol has two phases: *secure route discovery* and *secure data forwarding*. In the secure route discovery phase, a source node broadcasts a route request (RREQ) message to its neighbours. The RREQ message contains the sending node ID, the destination ID (the sink), an encrypted nonce and a MAC generated using the key shared with the sink node. Each intermediate node forwards the RREQ message to the next hop after updating the message with its ID and its previous hop ID. When the RREQ arrives at the sink, it verifies the MAC using the shared key, builds a route reply (RREP) message, and sends it out. The RREP message consists of the sink ID, source node ID, current intermediate node, and predecessor node protected by a MAC generated using the shared key. The routing tables of all the intermediate nodes are updated as the RREP message is propagated towards the source node. Upon receiving the RREP message, the source node verifies that the RREP message originated from the sink node using the MAC. If the RREP message is lost, possibly due to malicious intermediate nodes, the source node triggers another route discovery. After the route discovery phase completes, the sink transmits encrypted cluster group keys to nodes using the shared secret keys. The sensor nodes send encrypted data along with a MAC to their cluster heads using the cluster group key. Secure inter-cluster communication works similar to the route discovery phase.

The primary goal of this protocol is to provide secure data communication. This goal was achieved, however, at the expense of energy consumption. This routing protocol uses a naive clustering algorithm based on the

number of neighbours as the metric. This can lead to non-uniform clusters formation and nodes having the largest number of neighbours can die faster. A better clustering algorithm that can yield uniform clusters will be a useful improvement to the protocol. Moreover, the availability of the location information through GPS devices was not exploited in the formation of the load-balanced clustering. This protocol involves a huge amount of setup communication overhead which may dissipate the energy savings achieved by clustering.